NASA/CR-2005-213515

Final Report for the Development of the NASA Technical Report Server (NTRS)

Michael L. Nelson
Old Dominion University, Norfolk, Virginia

September 2005

NASA/CR-2005-213515

Final Report for the Development of the NASA Technical Report Server (NTRS)

Michael L. Nelson
Old Dominion University, Norfolk, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

Prepared for Langley Research Center
under Contract NAS1-01052

September 2005

Executive Summary

This report details the activity of the author under Modification Number 17 on NASA LaRC Contract NAS1-01052. The author performed a variety of research, development and consulting tasks for NASA Langley Research Center in the area of digital libraries (DLs) and supporting technologies, such as the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH).

In particular, the development focused on the NASA Technical Report Server (NTRS) and its transition from a distributed searching model to one that uses the OAI-PMH. The Open Archives Initiative (OAI) is an international consortium focused on furthering the interoperability of DLs through the use of "metadata harvesting". Metadata harvesting is in contrast to the “distributed searching” approach of many previous DL interoperability projects for federating different DLs into a single service. While feasible for small numbers of nodes (e.g., < 20), large-scale distributed searching has proven difficult in an Internet environment for large numbers of nodes (e.g., > 100).

The OAI-PMH version of NTRS went into public production on April 28, 2003. Since that time, it has been extremely well received. In addition to providing the NTRS user community with a higher level of service than the previous, distributed searching version of NTRS, it has provided more insight into how the user community uses NTRS in a variety of deployment scenarios.

This report details the design, implementation and maintenance of the NTRS. Source code is included in the appendices. URLs of the some of the frequently discussed resources in this report:

- NTRS: http://ntrs.nasa.gov/
- NACA: http://naca.larc.nasa.gov/
- LTRS: http://techreports.larc.nasa.gov/
- Open Archives Initiative: http://www.openarchives.org/
- Dublin Core: http://www.dublincore.org/
- Awstats: http://awstats.sourceforge.net/
- MySQL: http://www.mysql.com/

Contents

1. Introduction

The NASA Technical Report Server (NTRS) is now based on the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) (Nelson, Rocker & Harrison, 2003). This metadata harvesting version of NTRS represents a significant improvement in the distributed searching implementation of NTRS. The new version of NTRS represents the confluence of two interesting technologies: buckets (Nelson & Maly, 2001) and OAI-PMH (Lagoze, Van de Sompel, Nelson & Warner, 2002).

1.1 NASA DL History

NASA's history with web-based DLs dates back to 1993, when a WWW interface was provided for the Langley Technical Report Server (LTRS) (Nelson & Gottlich, 1994; Nelson, Gottlich & Bianco, 1994; Roper et al., 1994). Prior to this, LTRS was simply an anonymous FTP server that distributed technical reports authored and sponsored by NASA Langley Research Center. However, LTRS provided access to only reports from NASA Langley Research Center, and not other NASA centers and institutes. Beginning in 1994, software used to create LTRS was shared with other NASA installations and "LTRS-like" DLs were setup. However, it was not until 1995 that the NASA Technical Report Server (NTRS) was setup to provide integrated searching between the various NASA web-based DLs (Nelson et al., 1995). NTRS proved extremely successful with the general public and averaged approximately 30,000 search sessions per month. The total collection accessible through the distributed searching version of NTRS was approximately 4.5M abstracts and 300K full-text publications.

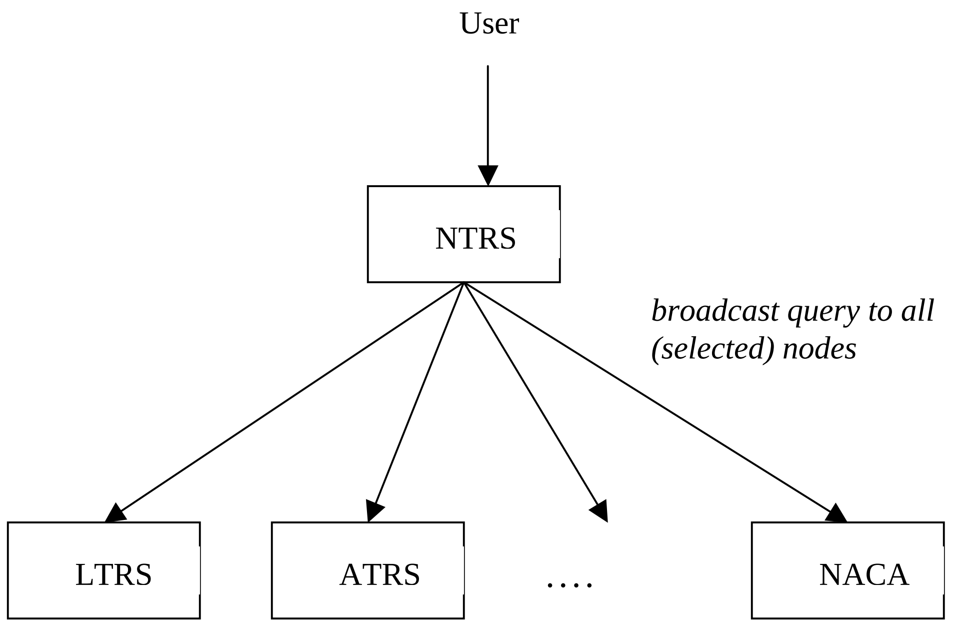
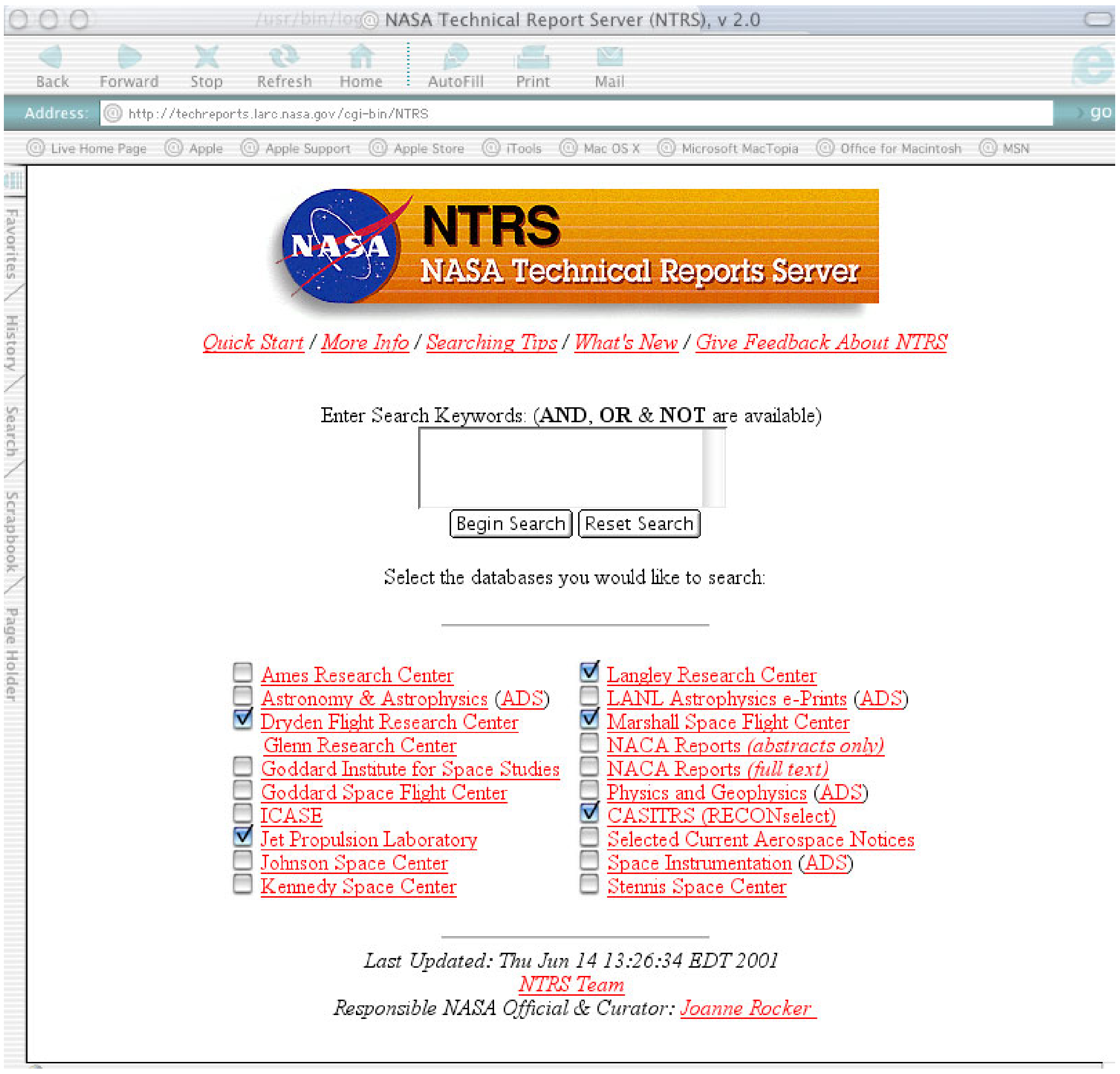


Figure 1. Architecture of the Distributed Searching NTRS



Quick Start / More Info / Searching Tips / What's New / Give Feedback About NTRS

Enter Search Keywords: (**AND**, **OR** & **NOT** are available)

Select the databases you would like to search:

[ ] Ames Research Center	[✓] Langley Research Center
[ ] Astronomy & Astrophysics (ADS)	[ ] LANL Astrophysics e-Prints (ADS)
[✓] Dryden Flight Research Center	[✓] Marshall Space Flight Center
Glenn Research Center	[ ] NACA Reports *(abstracts only)*
[ ] Goddard Institute for Space Studies	[ ] NACA Reports *(full text)*
[ ] Goddard Space Flight Center	[ ] Physics and Geophysics (ADS)
[ ] ICASE	[✓] CASITRS (RECONselect)
[✓] Jet Propulsion Laboratory	[ ] Selected Current Aerospace Notices
[ ] Johnson Space Center	[ ] Space Instrumentation (ADS)
[ ] Kennedy Space Center	[ ] Stennis Space Center

Last Updated: Thu Jun 14 13:26:34 EDT 2001
NTRS Team
Responsible NASA Official & Curator: Joanne Rocker

Figure 2. Screen Capture of the Distributed Searching Version of NTRS

The architecture for this version of NTRS is shown in Figure 1, and a screen capture of the user interface is shown in Figure 2. Integration of the 20 separate nodes comprising NTRS was done through distributed searching. A single Perl script generates the NTRS interface, processes and dispatches queries to a daemon specifically assigned to each NTRS node. The main script gathers the results from each of the daemons and presents the results to the user. No attempt is made to integrate the results from each node in any meaningful manner; the architecture of the institutional nodes was exposed to the user. Although the searching is done in parallel, generating the results will not run faster than the slowest component. If for some reason a component is not reachable, the search will be held up waiting for that search request to timeout. Thus for performance reasons, user queries are not broadcast to all 20 nodes by default, but rather only to six of the more popular nodes. 17 of the nodes support different versions of WAIS and the three nodes from the Astrophysics Data Service utilize a non-WAIS version of Z39.50 (Kurtz et al., 1999). Popular in the early days of the WWW, WAIS (Kahle et al., 1992) is an Internet based subset of the venerable Z39.50 standard for distributed information retrieval (Lynch, 1997).

While the distributed searching version of NTRS was popular and useful, its distributed searching architecture imposed many limitations. Since NTRS did not contain holdings of it own, it was dependent on the quality and availability of its constituent nodes. As has been reported for similar distributed searching DLs, the availability of all nodes in a distributed searching environment can be poor (Powell & French, 2000). Furthermore, syntactical differences between the various search engines limits the complexity of the searches that can be specified. While NTRS did some search syntax normalization, the mappings provided are less than complete.

1.2 Adopting the OAI-PMH to Existing NASA DLs

NASA was one of the original participants in the OAI. For the Universal Preprint Service (UPS) demonstration project that lead to the creation of the OAI (Van de Sompel et al., 2000), NASA contributed the metadata from the National Advisory Committee for Aeronautics (NACA) digital library. The NACA digital library is a retrospective digitization project that focuses on the capture and preservation of the reports authored by NASA's predecessor organization during 1917-1958 (Nelson, 1999). In late 2001, both LTRS and NACA had OAI-PMH interfaces available for harvesting. This allowed the content in both collections to be indexed by a variety of OAI service providers While LTRS and NACA had previously been crawled by web crawlers and thus had their contents available from general search engines (e.g., Google, AltaVista, AskJeeves, etc.), the OAI-PMH interface was the first time that these DLs were truly interoperable with generalized services.

Adding OAI-PMH interfaces to the existing DLs took only a few days of programming for each interface. In fact, after a generalized approach was developed, several OAI-PMH interfaces for other nodes in NTRS were developed by the authors and distributed in less than a day. The approach used in these implementations was to add a modified "bucket" to the DLs. Buckets are object-oriented, intelligent web storage units that contain metadata, data and methods to operate on the metadata and date (Nelson & Maly, 2001). Buckets already implement approximately 30 methods (or "verbs" in OAI-PMH parlance), so adding the six OAI-PMH verbs as bucket methods was easily accomplished. Buckets were originally built to hold full-text content in DLs, but they have proved useful in other scenarios as well, including building OAI-PMH repositories. Figure 3 shows the architecture of the modified bucket. The six OAI-PMH verbs are added to the "package" that contains the other bucket methods. These OAI-PMH verbs are implemented generically; they depend on a library file that maps the general implementations of the OAI-PMH verbs to the explicit DL. For example, it is in this file where native metadata formats are mapped to Dublin Core, rules for constructing identifiers are kept, and the appropriate filesystem or SQL calls are defined to extract the necessary metadata. Table 1 is a list of all the NASA OAI-PMH repositories built using modified buckets.

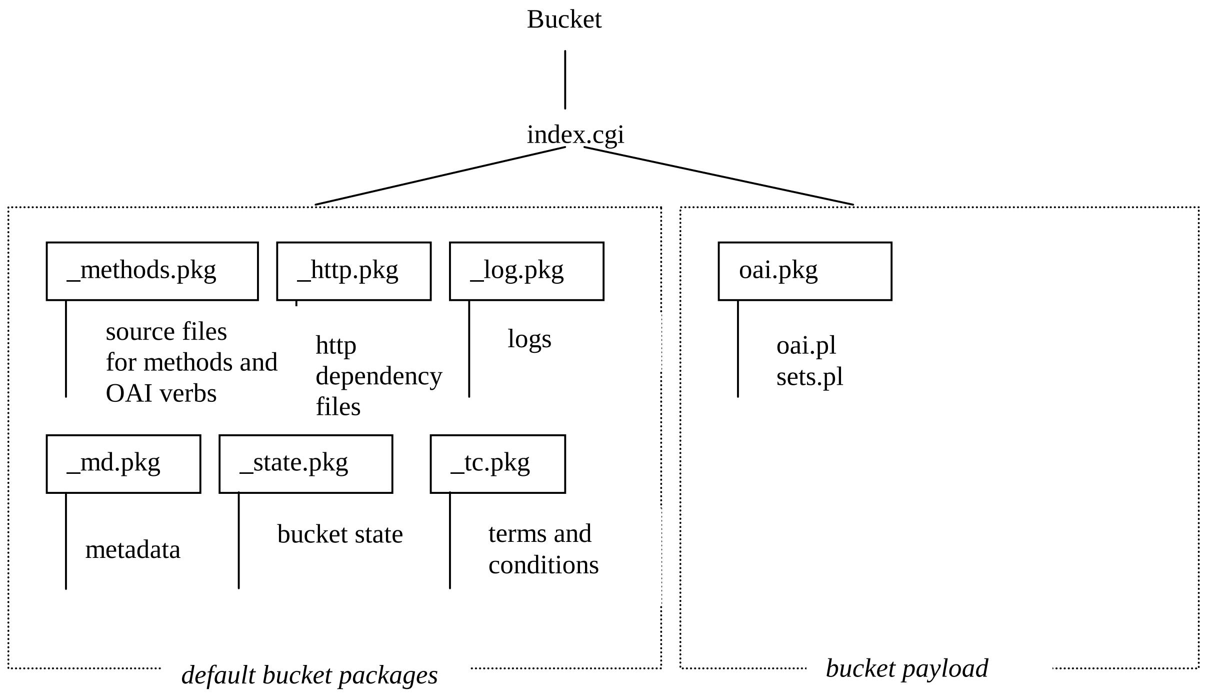


Figure 3. Architecture of an OAI-PMH Bucket

Name	URL	Notes
Langley Technical Report Server	http://techreports.larc.nasa.gov/ltrs/oai/ (1.1) http://techreports.larc.nasa.gov/ltrs/oai2.0/ (2.0)	Metadata in refer format; stored on a filesystem
NACA	http://naca.larc.nasa.gov/oai/ (1.1) http://naca.larc.nasa.gov/oai2.0/ (2.0)	Metadata in refer format; stored on a filesystem
Johnson Space Center Technical Report Server	http://ston.jsc.nasa.gov/JSCTRS/oai/ (1.1)	Metadata in a COSATI derivative; stored in a MS Access database dump
Goddard Institute for Space Science	http://www.giss.nasa.gov/cgi-bin/gpol2/ (2.0)	Metadata in a locally defined format; stored in MySQL

Table 1. OAI-PMH Interfaces Built With Buckets

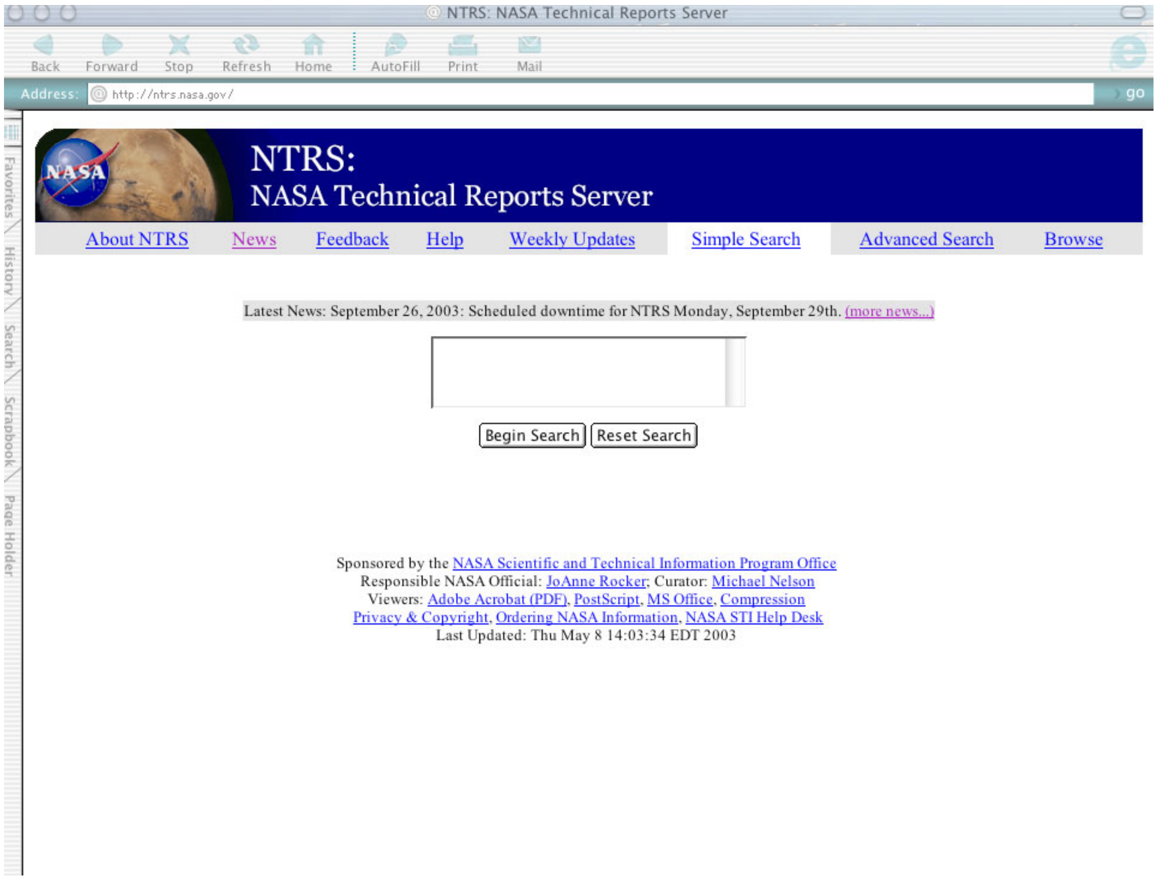
LTRS and NACA currently see a great deal of OAI-PMH harvesting activity. Some of this is surely due to spikes of harvesting and testing as new SPs become active. However, the breadth of activity (over 20 distinct harvesters have visited LTRS and NACA) indicates significant acceptance of OAI-PMH interfaces for these DLs. These DLs still see the same amount of traffic from interactive users and regular web crawlers, but the OAI-PMH capabilities result in even more exposure for their contents.

1.3 An OAI-PMH Version of NTRS

The new NTRS offers many advantages that the earlier, distributed searching NTRS does not. For one, all the contents of NTRS are now searched by default. In the previous version of NTRS, not all nodes were searched by default because searching all nodes is "expensive" in terms of network resources. Furthermore, many of the nodes are highly specialized and are less likely to produce interesting results for the general query. In an attempt to limit the penalty of network access to nodes that are not likely to produce interesting results, only 6 of the 20 nodes selected by default. Of course, depending on knowledgeable users to correctly target their searches to the right nodes is an unrealistic assumption made by the previous NTRS version. However, this is not an issue in the new version of NTRS because metadata harvesting means maintaining a copy of metadata harvested in batch mode and not having to dynamically search each node for every query. This results in faster, more reliable searches for users. NTRS currently does not cache copies of full-text documents, so full-text document availability is still subject to transient network errors. The new version of NTRS provides both a simple interface (Figure 4) and an advanced search interface (Figure 5) that allows more targeted searching, including limiting the number of repositories to search. Syntactic differences between the 20 nodes of the previous version of NTRS made it infeasible to offer anything beyond just a simple search interface.

New in the OAI-PMH version of NTRS is the inclusion of repositories that are not in the nasa.gov domain. At the moment, we include repositories from the Physics eprint Server (arXiv), Biomecentral (a free biomedical journal publisher), reports from the UK-equivalent of NASA (the Aeronautical Research Council) and a test database from the Department of Energy's Office of Scientific and Technical Information (OSTI). The simple search interface searches only the NASA repositories by default. The advanced search interface (which features fielded searching) offers the possibility of including non-NASA repositories. An important distinction is that the non-NASA repositories are not selected by default, the user doing an advanced search must explicitly request non-NASA repositories to be included in the search.

We are actively recruiting some of the NASA centers not currently represented in NTRS to join in the near future. For example, the NASA Jet Propulsion Laboratory (JPL) is not currently a participant in the OAI-PMH version of NTRS, but will be in the near future. At the moment, JPL is re-engineering its knowledge management processes and using OAI-PMH as a core technology for this purpose. When this project is complete, an OAI-PMH interface for its technical publications (suitable for public use in NTRS) will be a small subset of JPL's institutional OAI-PMH deployment. We have also submitted code for Glenn Research Center to join NTRS and we await the results of their effort. We are initiating discussion with Dryden Flight Research Center. We are unsure of the status of Ames Research Center, Goddard Space Flight Center, Kennedy Space Center and Stennis Space Center.



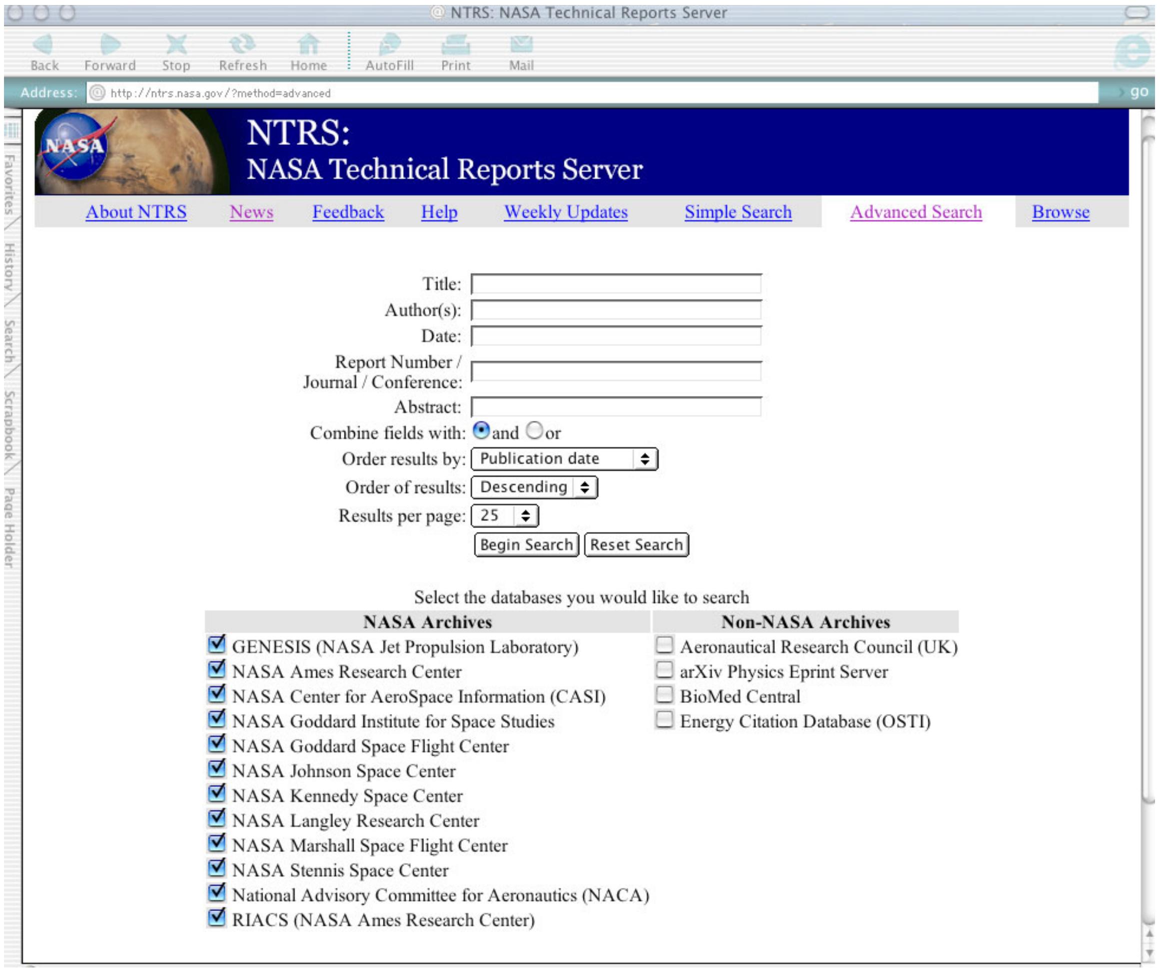
NTRS:
NASA Technical Reports Server

About NTRS News Feedback Help Weekly Updates Simple Search Advanced Search Browse

Latest News: September 26, 2003: Scheduled downtime for NTRS Monday, September 29th. (more news...)

Sponsored by the NASA Scientific and Technical Information Program Office
Responsible NASA Official: JoAnne Rocker; Curator: Michael Nelson
Viewers: Adobe Acrobat (PDF), PostScript, MS Office, Compression
Privacy & Copyright, Ordering NASA Information, NASA STI Help Desk
Last Updated: Thu May 8 14:03:34 EDT 2003

Figure 4. Simple Search Interface of the OAI-PMH NTRS



Title:	
Author(s):	
Date:	
Report Number / Journal / Conference:	
Abstract:	
Combine fields with:	[✓ and] or
Order results by:	Publication date
Order of results:	Descending
Results per page:	25

Select the databases you would like to search

NASA Archives	Non-NASA Archives
✓ GENESIS (NASA Jet Propulsion Laboratory)	☐ Aeronautical Research Council (UK)
✓ NASA Ames Research Center	☐ arXiv Physics Eprint Server
✓ NASA Center for AeroSpace Information (CASI)	☐ BioMed Central
✓ NASA Goddard Institute for Space Studies	☐ Energy Citation Database (OSTI)
✓ NASA Goddard Space Flight Center	
✓ NASA Johnson Space Center	
✓ NASA Kennedy Space Center	
✓ NASA Langley Research Center	
✓ NASA Marshall Space Flight Center	
✓ NASA Stennis Space Center	
✓ National Advisory Committee for Aeronautics (NACA)	
✓ RIACS (NASA Ames Research Center)	

Figure 5. Advanced Search Screen of the OAI-PMH NTRS

One interesting feature now offered by the OAI-PMH version of NTRS is that of "hierarchical harvesting" of the kind first introduced by Arc (Liu, Maly, Zubair & Nelson, 2001). Hierarchical harvesting is gaining popularity as OAI-PMH aggregators harvest metadata from data providers and re-expose the metadata to other harvesters, perhaps with some metadata normalization or cleansing. This means that NTRS can re-export its contents and act as an aggregator of public NASA STI. For example, the Elsevier sponsored OAI SP, Scirus, currently harvests both LTRS and NACA as separate repositories, even though they belong to a larger logical grouping. However, once the OAI-PMH version of NTRS is online, Scirus will be able to harvest a single NASA repository and have the entirety of the harvestable public NASA STI. Similarly, we are in discussions with the science.gov effort and hope to demonstrate the feasibility of an OAI-PMH hierarchical harvesting approach for their project. Hierarchical harvesting is a powerful capability that will make the OAI-PMH flexible enough to allow DLs to adapt to organizational structures. Guidelines for hierarchical harvest (known as "aggregators" in OAI-PMH parlance) can be found at (Lagoze, Van de Sompel, Nelson & Warner, 2002).

The new NTRS is itself a modified bucket, much like an extension of the OAI-PMH interfaces for LTRS and NACA presented above. The NTRS bucket has the standard bucket methods, the OAI-PMH verbs (for hierarchical harvesting), and a set of NTRS specific verbs to correspond to simple and advanced search interfaces, the actual searching functions, etc. Figure 6 illustrates the architecture of the NTRS bucket. The metadata for NTRS is harvested using the harvester developed as part of the Open Digital Library Project at Virginia Tech (Suleman & Fox, 2001). The metadata is stored on a file system in its harvested form, then a locally developed Perl script loads the metadata into a MySQL database. Currently, only Dublin Core metadata (Weibel & Lagoze, 1997) is harvested from the participating nodes.

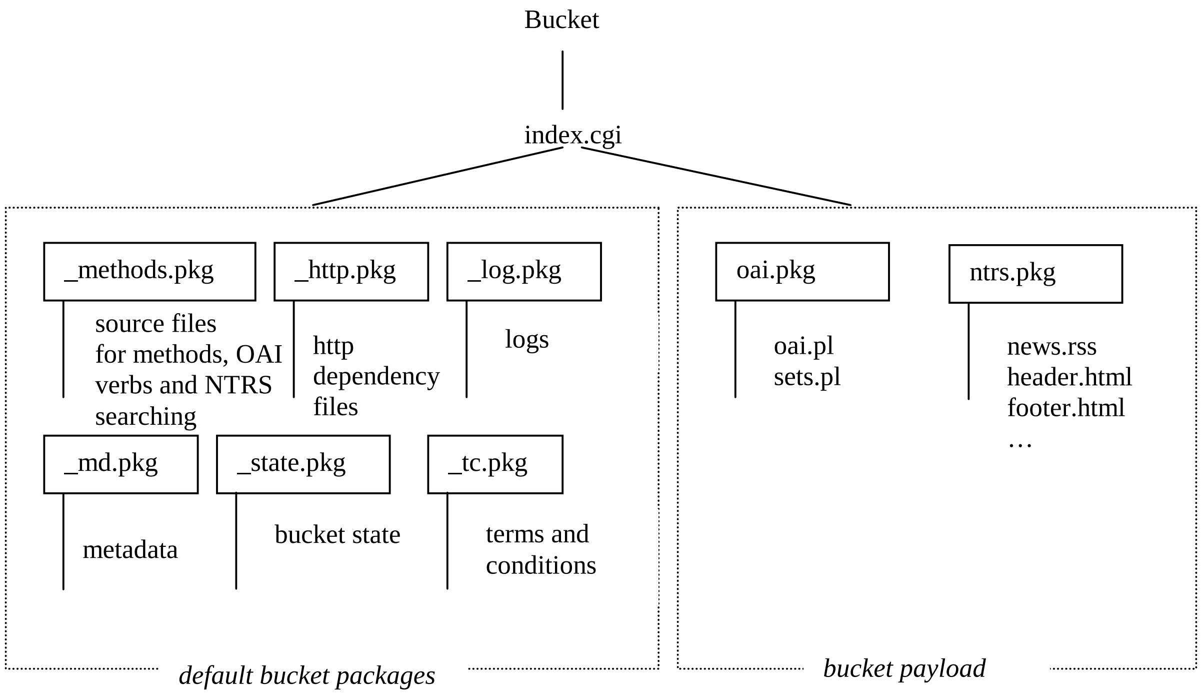


Figure 6. Architecture of the NTRS Bucket

2. Bucket Architecture

In this section, we explore in detail the architectural issues introduced in section 1. We take a tour of the NTRS implementation, with a file-by-file examination. NTRS is hosted on a belvedere.larc.nasa.gov, a Sun Sparcstation:

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % uname -X
System = SunOS
Node = belvedere
Release = 5.9
KernelID = Generic_112233-06
Machine = sun4u
```

```
BusType = <unknown>
Serial = <unknown>
Users = <unknown>
OEM# = 0
Origin# = 1
NumCPU = 2
belvedere:/usr/local/web/htdocs/ntrs % df -k
Filesystem            kbytes    used   avail capacity  Mounted on
/dev/dsk/c0t0d0s0    1016122   88905  866250    10%    /
/dev/dsk/c0t0d0s3    2053605 1129790  862207    57%    /usr
/proc                      0       0       0     0%    /proc
mnttab                     0       0       0     0%    /etc/mnttab
fd                         0       0       0     0%    /dev/fd
/dev/dsk/c0t0d0s5    7226450  654383 6499803    10%    /var
swap                 6935216      40 6935176     1%    /var/run
swap                 6935184       8 6935176     1%    /tmp
/dev/dsk/c0t0d0s4    8258597  383576 7792436     5%    /opt
/dev/dsk/c0t0d0s7    4129290 2756079 1331919    68%    /export/home
/dev/dsk/c0t0d0s6    8167589 7775493  310421    97%    /usr/local
/dev/dsk/c3t0d0s7    35004827 24415994 10238785    71%
/usr/local/web/htdocs/naca
/dev/dsk/c3t1d0s7    35004827 29828228 4826551    87%
/usr/local/web/htdocs/naca1
/dev/dsk/c3t3d0s7    35004827 15180323 19474456    44%
/usr/local/web/htdocs/naca3
/dev/dsk/c3t2d0s7    35004827 29926415 4728364    87%
/usr/local/web/htdocs/naca2
/dev/dsk/c2t0d0s7    35004827 8095133 26559646    24%
/usr/local/web/htdocs/ntrs
/dev/dsk/c2t1d0s7    35004827       9 34654770     1%
/usr/local/web/htdocs/ntrs1
/dev/dsk/c2t2d0s7    35004827 34479539  175240   100%
/usr/local/web/htdocs/ntrs2
```

2.1 Web Configuration

ntrs.nasa.gov is configured as a virtual server on belvedere. The http server is Apache:

```
belvedere:/usr/local/apache % bin/httpd -v
Server version: Apache/1.3.26 (Unix)
Server built:   Oct 18 2002 13:28:14
```

Some relevant snippets from the apache configuration file:

```
belvedere:/usr/local/apache % less /etc/apache/httpd.conf
…
#
# This may also be "None", "All", or any combination of "Indexes",
# "Includes", "FollowSymLinks", "ExecCGI", or "MultiViews".
#
# Note that "MultiViews" must be named *explicitly* --- "Options All"
# doesn't give it to you.
#
    Options Indexes FollowSymLinks MultiViews ExecCGI

#
# This controls which options the .htaccess files in directories can
# override. Can also be "All", or any combination of "Options", "FileInfo",
# "AuthConfig", and "Limit"
#
    AllowOverride All
…
#
# DirectoryIndex: Name of the file or files to use as a pre-written HTML
# directory index.  Separate multiple entries with spaces.
```

```
#
<IfModule mod_dir.c>
    DirectoryIndex index.html index.cgi
</IfModule>
…
NameVirtualHost 128.155.4.34
<VirtualHost 128.155.4.34>
    ServerAdmin m.l.nelson@larc.nasa.gov
    DocumentRoot /usr/local/web/htdocs/ntrs/ntrs
    ServerName ntrs.nasa.gov
    ErrorLog /var/web/ntrs/error
    CustomLog /var/web/ntrs/access combined
</VirtualHost>
```

2.2 Filesystem Configuration

Note from above that the httpd DocumentRoot is /usr/local/web/htdocs/ntrs/ntrs. However, we begin our tour in the parent director; /usr/local/web/htdocs/ntrs/ntrs is explored in section 2.3.

```
belvedere:/usr/local/web/htdocs/ntrs % ls
awstats-5.4/      lost+found/       ODL-Harvest/      src/
bin/              ntrs/             ODL-Harvest-2.0/
```

2.2.1 src

The “lost+found” directory is a side-effect of /usr/local/web/htdocs/ntrs being its own Unix filesystem partition. The “src” directory contains tar files of some of the software that has been installed on belvedere to have NTRS run correctly:

```
belvedere:/usr/local/web/htdocs/ntrs % ls src
awstats-5.4.tgz
bison-1.875.tar.gz
Data-Dumper-2.101/
Data-Dumper-2.101.tar.gz
DBI-1.18/
DBI-1.18.tar.gz
Harvest-1.11.tar.gz
Harvest-2.0.tar.gz
m4-1.4.tar.gz
make-3.80.tar.gz
Msql-Mysql-modules-1.2216/
Msql-Mysql-modules-1.2216.tar.gz
mysql-3.23.49-sun-solaris2.8-sparc.tar.gz
mysql-4.0.12.tar.gz
mysql-4.0.5.tar.gz
tar-1.13/
tar-1.13.tar.gz
```

“awstats” is a freely available httpd log file analysis package; it has its own directory at /usr/local/web/htdocs/ntrs (see section 4.3 for more information). “Data-Dumper” and “Msql-Mysql-modules” are required for the installation of “DBI”, a software package that provides allows for easy access of database systems from Perl (similar in spirit to “JDBC” for Java). “bison”, “make”, “m4”, and “tar” are GNU replacements for the incorrectly functioning utilities of the same name (substitute “bison” for “yacc”) that are standard in Solaris. They are required to recompile MySQL (section 2.4). We have used several different versions of the "MySQL" relational database management systems (RDBMS) in NTRS.

2.2.2 ODL-Harvest

The two versions of "ODL-Harvest" map to versions 1.1 and 2.0 of the OAI-PMH, respectively. These are the OAI-PMH Harvesters from Virginia Tech (Suleman & Fox, 2001). Since not all repositories in NTRS are using OAI-PMH 2.0, we support both versions of OAI-PMH, and thus, ODL-Harvest. However, to integrate the metadata directories into the same parent directory, we use symbolic links to map the repository name from ODL-Harvest to ODL-Harvest-2.0:

```
belvedere:/usr/local/web/htdocs/ntrs % ls ODL-Harvest/Harvest/
arc/                    genesis.jpl.nasa.gov@  magicnrc@
arXiv.org@              gtrs/                  mtrs/
arxiv.org@              gtrs.gsfc.nasa.gov@    naca.larc.nasa.gov@
atrs/                   harvest-all*           naca.tar.gz
atrs.arc.nasa.gov@      icase/                 NASAMSFCEPRINTS@
bad.records             JTRS@                  RIACS@
biomedcentral.com@      jtrs/                  riacs/
casi/                   ktrs/                  ssctrs/
casi.ntrs.nasa.gov@     ktrs.ksc.nasa.gov@     ssctrs.ssc.nasa.gov@
configure.pl*           ltrs.larc.nasa.gov@    template/
ecd.osti.gov@           ltrs.tar.gz            www.giss.nasa.gov@
belvedere:/usr/local/web/htdocs/ntrs % ls ODL-Harvest-2.0/Harvest/
arxiv.org/              genesis.jpl.nasa.gov/  template/
biomedcentral.com/      harvest-all*           www.giss.nasa.gov/
configure.pl*           ltrs.larc.nasa.gov/
ecd.osti.gov/           naca.larc.nasa.gov/
```

The names of the subdirectories listed above correspond to the repository names used in the NTRS interface.

2.2.3 bin

The "bin" directory has a number of support scripts necessary for the operation of NTRS:

```
belvedere:/usr/local/web/htdocs/ntrs % ls bin
archives-html.pl    entities.pl         log-cron            usage.pl
build-table.pl      harvest-cron        populate-ntrs.pl
delete.pl           harvest-cron.old    today.pl
```

The source code for these scripts is included in the appendices. "archives-html.pl" builds HTML snippets for inclusion in the NTRS displays based on the values in the "archives" database table. This prevents the NTRS administrator from having to rebuild the HTML display when new repositories are added. "build-table.pl" creates the tables used in NTRS (and can be used to reset the NTRS database to a null state). "delete.pl" is a helper script to delete particular files or entire repositories from the database. "entities.pl" is used to process XML files. "harvest-cron" is weekly script for invoking the ODL-Harvest utility to harvest, using OAI-PMH, the known repositories. "log-cron" is a monthly script that is run from cron to process the log files and create usage summaries. “populate-ntrs.pl” is used to index the harvest XML metadata into the MySQL database. “today.pl” returns the current day in YYYY-MM-DD format (used in a number of other scripts). “usage.pl” is a helper script that is called from “harvest-cron” that is used to prepare usage summaries suitable for the “analyze” bucket method.

2.2.4 Crontab

The values that are used in the crontab entry are:

```
belvedere:/usr/local/web/htdocs/ntrs % crontab -l
1 19 * * 5 /usr/local/web/htdocs/ntrs/bin/harvest-cron
0 1 1 * * /usr/local/web/htdocs/ntrs/bin/log-cron
```

Currently these values are in the crontab entry for user “mln”. If this user id were to be deleted, the crontab entries would have to be moved to another user’s crontab. “harvest-cron” is run weekly at 7:01 pm every Friday. “log-cron” is run at 1 am on the first of every month.

2.3 Bucket Configuration

In this section, we discuss the structure of the “ntrs” subdirectory in the “/usr/local/web/htdocs/ntrs” directory (section 2.2). As mentioned in section 1.3, NTRS is actually implemented as a bucket. However, there are many extensions and additional capabilities in this bucket that separate from its simpler data preservation oriented brethren. The default bucket method is no longer “display”, but is “simple”. Since basic bucket functionality is documented elsewhere (Nelson, 2000), we will focus on the functions that are unique to NTRS. The version of the bucket used to implement NTRS is 1.6.3.

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls
archive/       images/        _methods.pkg/  ntrs.tar.gz    _state.pkg/
casi@          index.cgi*     naca@          oai/           _tc.pkg/
casi.tar.gz    _log.pkg/      naca.tar.gz    redirect*
_http.pkg/     _md.pkg/       ntrs.pkg/      restricted@
```

“casi”, “casi.tar.gz”, “naca”, “naca.tar.gz”, “ntrs.tar.gz” and “redirect” are vestiges of testing, set-up and transition and can be ignored. They are likely to be deleted in the near future.

2.3.1 images

The “images” directory is not protected by a “.htaccess” file like most subdirectories in a bucket. This is eliminated to reduce the overhead associated with accessing files in buckets. This allows URLs of the type:

http://ntrs.nasa.gov/images/mast.gif

to be used instead of the more standard bucket mechanism of retrieving an element from a package:

http://ntrs.nasa.gov/?method=display&pkg_name=images&element_name=mast.gif

Since these are just navigational and cosmetic images, the overhead of the latter URL is not justified.

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls images/
colinsig.gif        icon/               z2.gif              z_logo.png
hotnasa2.gif        insignia64x53.gif   z2.png              z.png
hotnasa3.gif        mast.gif            z.gif
hotnasa.gif         meatballclr.gif     z_logo.gif
```

2.3.2 archive

The “archive” directory contains the metadata and data of both NASA sites that have no plans to join NTRS in the immediate future (ARC, GSFC, KSC, SSC) as well as a mirror of the MAGiC project UK Aeronautical Research Council (ARC) reports (Sidwell, Needham &Harrington, 2000). Since the file listing is quite long, only the directories are shown below:

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls archive/
nasa/      non-nasa/
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls archive/nasa/
atrs.arc.nasa.gov/    hqtrs.hq.nasa.gov/    ssctrs.ssc.nasa.gov/
casi.ntrs.nasa.gov/   ktrs.ksc.nasa.gov/
gtrs.gsfc.nasa.gov/   runme.pl*
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls archive/non-nasa/
uk/
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls archive/non-nasa/uk/arc/
cp/   rm/
```

2.3.3 oai

The “oai” directory contains the information necessary to respond to OAI-PMH requests, thus making NTRS an OAI-PMH aggregator. The functionality of this approach has been described elsewhere (Nelson, Rocker, Harrison, 2003), but the source code is included in the appendices.

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls oai
oai.pl    sets.pl
```

The base URL for OAI-PMH harvesting is http://ntrs.nasa.gov/. Implementing NTRS as a bucket makes it easy to overload the same URL for human users as well as harvesters.

2.3.4 ntrs

The “ntrs” directory contains a number of support files, including “news.rss”, an XML encoded news feed in the Rich Site Summary (RSS) format used in the “news” bucket method; the headers and footers that appear on every NTRS page; the output of the archives-html.pl program (“checkbox.html”, “radio.html”, and “select.html”) – which provides an automatic mechanism for generated HTML tables for all the various NTRS repositories; “admin.html”, which is used to generate the HTML display in the “admin” bucket method; as well as the support and data files for the automatic survey mechanism contributed by William Von Ofenheim (NASA LaRC).

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls ntrs.pkg/
admin.html~         header2.html        news.rss            survey-support.pl
admin.html          header.html         radio.html
checkbox.html       header.html.new     select.html
footer.html         header.html.old     survey/
```

2.3.5 methods

The standard bucket API is fully documented in Nelson (2000). The listing below gives all the methods defined in the NTRS bucket, with the OAI-PMH methods highlighted in blue, and the NTRS-RDBMS methods highlighted in red. The source code for all the new methods is included in the appendices. Alternate or test versions of the methods listed below are ignored.

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % ls _methods.pkg/
about.pl                 feedback.pl              metadata.pl
add_element.pl           get_log.pl               news.pl
add_method.pl            get_preference.pl        ordering.pl
add_package.pl           GetRecord.pl             pack.pl
add_principal.pl         get_state.pl             privacy.pl
add_tc.pl                growth.pl                recommendation.pl
admin.pl                 help.pl                  results.pl
advanced.pl              Identify.pl              search2.pl
analyze.pl               id.pl                    search.pl
analyze.pl.orig          lint.pl                  search.pl.orig
browse.pl                ListIdentifiers.pl       set_metadata.pl
delete_bucket.pl         list_logs.pl             set_preference.pl
delete_element.pl        ListMetadataFormats.pl   set_state.pl
delete_log.pl            list_methods.pl          set_version.pl
delete_method.pl         list_principals.pl       simple.pl
delete_package.pl        ListRecords.pl           unpack.pl
delete_principal.pl      ListSets.pl              updates.pl
delete_tc.pl             list_source.pl           version.pl
display.pl               list_tc.pl
```

Most of the NTRS specific functions are very obvious in their goal (e.g., “news”, “ordering”, etc.). One subtle factor that bears mention is that only the “search” method actually contacts the MySQL database to perform searches; the HTML interfaces generated by the “simple”, “advanced”, “browse”, and “updates” methods all have as their ACTION (in the HTML Form) the “search” method. A “mode” variable is set so that the “search” method knows the originating method and frames the actual search accordingly. An inspection of the code for the search method will make this clearer.

2.3.6 Terms and Conditions

Many of the methods have terms and conditions associated with them. This prevents unauthorized people from performing privileged actions on the NTRS bucket. Using the standard bucket method of "list_tc", we can see the following values returned:

http://ntrs.nasa.gov/?method=list_tc

```
add_element:
user:marks
user:mln
user:joanne
add_method:
user:marks
user:mln
user:joanne

add_package:
user:marks
user:mln
user:joanne

add_principal:
user:marks
user:mln
user:joanne

add_tc:
user:marks
user:mln
user:joanne

delete_bucket:
user:marks
user:mln
user:joanne

delete_element:
user:marks
user:mln
user:joanne

delete_log:
user:marks
user:mln
user:joanne

delete_method:
user:marks
user:mln
user:joanne

delete_package:
user:marks
user:mln
user:joanne

delete_principal:
user:marks
user:mln
user:joanne

set_metadata:
user:marks
user:mln
user:joanne

set_state:
```

```
user:marks
user:mln
user:joanne

set_version:
user:marks
user:mln
user:joanne

unpack:
user:marks
user:mln
user:joanne

get_log:
user:marks
user:mln
user:joanne

admin:
user:marks
user:mln
user:joanne
user:george
user:jack
user:calvin
user:laura
user:lauretta

display:
package:_log.pkg
user:marks
user:mln
user:joanne
user:george
user:jack
user:calvin
user:laura
user:lauretta

analyze:
user:marks
user:mln
user:joanne
user:george
user:jack
user:calvin
user:laura
user:lauretta
```

The password file for these principals can be seen by:

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % more _tc.pkg/passwd
mln:saf7fujRtiBGk
joanne:saBcQ1bdjiNic
marks:saa8G.84oA4I.
jack:sa83q7F7u.Sd.
george:sadtaN6fNsCkw
calvin:sadXS.crma8wk
laura:saLR04PdNjN/c
lauretta:saLVWP0mum/6I
```

The passwords are encrypted in standard Unix fashion. These values can be modified via the bucket “admin” method (section 4.2). The HTML displays generated by each of the NTRS-specific methods are shown in the appendices.

2.4 MySQL Configuration

We use MySQL version 4.0.12. Version 4.x provides the full-text indexing necessary for the type of searches that NTRS performs. The MySQL server is started at run-time with:

```
belvedere:/usr/local/web/htdocs/ntrs/ntrs % less /etc/rc3.d/S81safemysqld
#! /bin/sh

#Start sql process for Michael Nelson - G.W.K and W.E.B 07/09/02
#added "--ft_min_word_len=3" - MLN 12/10/02
#added "--ft_min_word_len=1" and changed to 4.x syntax - MLN 04/24/03
#added changed query_cache_size to 400000000 - MLN 08/14/03
#changed safe_mysqld to mysqld_safe - MLN 09/24/03

if test -x /usr/local/mysql/bin/mysqld_safe ; then
        /usr/local/mysql/bin/mysqld_safe --user=mysql --ft_min_word_len=1 --
query_cache_size=400000000 &
fi

exit 0
```

A number of modifications are shown above. First, we use a query cache of .4 gigabytes to increase performance. Second, we set the minimum word length to 1; the default is value 4. Unfortunately, a default world length of 4 causes surnames such as “Liu” and acronyms such as “CFD” to be treated as stop words. MySQL had to be recompiled with a new stopword list; myisam/ft_static.c was edited to remove "zero", "one", "two" .. "nine" from the stopword list. For non-technical communication these are good stopwords, but it caused titles to be skipped that contained phrases like “three-degrees of freedom”.

If the MySQL server dies, the “search” method will send an email to ntrs+admin@larc.nasa.gov when it discovers that the server is down.

3. Database Schema

NTRS is implemented as a set of three tables in a MySQL database. The “archives” table contains information describing from where the metadata was harvested. The “harvests” metadata contains information pertaining to the number of records of a particular archive at different times during the past. The “metadata” table contains the Dublin Core metadata as well as some of the OAI-PMH response header information. Even though all 15 Dublin Core fields are included

The following is a log of a MySQL console session that shows the complete database schema for NTRS.

```
%mysql -u mln -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3855 to server version: 4.0.12-log

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use ntrs;
```

```
Database changed
mysql> describe metadata;
+-------------+---------+------+-----+------------+-------+
| Field       | Type    | Null | Key | Default    | Extra |
+-------------+---------+------+-----+------------+-------+
| oaiID       | text    |      | PRI |            |       |
| dateStamp   | date    |      | MUL | 0000-00-00 |       |
| archiveID   | int(11) |      | MUL | 0          |       |
| title       | text    | YES  | MUL | NULL       |       |
| creator     | text    | YES  | MUL | NULL       |       |
| subject     | text    | YES  |     | NULL       |       |
| description | text    | YES  | MUL | NULL       |       |
| publisher   | text    | YES  |     | NULL       |       |
| contributor | text    | YES  |     | NULL       |       |
| date        | text    | YES  | MUL | NULL       |       |
| type        | text    | YES  | MUL | NULL       |       |
| format      | text    | YES  |     | NULL       |       |
| identifier  | text    | YES  |     | NULL       |       |
| source      | text    | YES  |     | NULL       |       |
| language    | text    | YES  |     | NULL       |       |
| relation    | text    | YES  |     | NULL       |       |
| coverage    | text    | YES  |     | NULL       |       |
| rights      | text    | YES  |     | NULL       |       |
+-------------+---------+------+-----+------------+-------+
18 rows in set (0.00 sec)

mysql> describe archives;
+------------+---------+------+-----+---------+-------+
| Field      | Type    | Null | Key | Default | Extra |
+------------+---------+------+-----+---------+-------+
| archiveID  | int(11) |      | PRI | 0       |       |
| shortName  | text    | YES  |     | NULL    |       |
| longName   | text    | YES  |     | NULL    |       |
| nasa       | text    | YES  |     | NULL    |       |
| url        | text    | YES  |     | NULL    |       |
| numRecords | int(11) | YES  |     | NULL    |       |
+------------+---------+------+-----+---------+-------+
6 rows in set (0.00 sec)

mysql> describe harvests;
+-----------+---------+------+-----+------------+-------+
| Field     | Type    | Null | Key | Default    | Extra |
+-----------+---------+------+-----+------------+-------+
| day       | date    |      | PRI | 0000-00-00 |       |
| archiveID | int(11) |      | PRI | 0          |       |
| size      | int(11) |      | PRI | 0          |       |
+-----------+---------+------+-----+------------+-------+
3 rows in set (0.07 sec)
```

4. Operating NTRS

Most NTRS operations can be performed through the web interface that it provides. However, some tasks require command line interaction.

4.1 Adding a Repository

Adding a new repository requires command line access and is one of the more complex operations required. Fortunately, it is a relatively rare occurrence and should not impose too much of a load on the NTRS administrator. The administrator must configure the OAI-PMH harvester, edit a Perl script, and run another script to update the HTML displays.

4.1.1 Harvester

The ODL-Harvester must be configured. Below is a log of what is required to add the repository naca.larc.nasa.gov. We always choose the default values.

```
belvedere:/usr/local/web/htdocs/ntrs/ODL-Harvest-2.0/Harvest % ./configure.pl
naca.larc.nasa.gov
+------------------------------------------+
| Harvester Sample Configurator            |
+------------------------------------------+
| Version 1.1 :: July 2002                 |
| Hussein Suleman <hussein@vt.edu>         |
| Digital Library Research Laboratory      |
| www.dlib.vt.edu :: Virginia Tech         |
------------------------------------------+

Defaults/previous values are in brackets - press <enter> to accept those
enter "&delete" to erase a default value
enter "&continue" to skip further questions and use all defaults
press <ctrl>-c to escape at any time (new values will be lost)

Press <enter> to continue

[ARCHIVES]
Add all the archives that should be harvested

Current list of archives:
No archives currently defined !

Select from: [A]dd   [D]one
Enter your choice [D] : a

[ARCHIVE IDENTIFIER]
You need a unique name by which to refer to the archive you
will harvest metadata from
Examples: naca.larc.nasa.gov, VTETD

Archive identifier [] : naca.larc.nasa.gov

[ARCHIVE URL]
This is the baseURL of the archive being harvested from.

Archive baseURL [] : http://naca.larc.nasa.gov/oai2.0/
Issuing Identify request to archive ...
Got response code : 200

[HARVESTING INTERVAL]
Choose how often you want to harvest (in seconds)
86400 = daily, 3600 = hourly

Harvesting interval [86400] :

[HARVESTING OVERLAP]
You need to select the number of seconds by which to overlap
harvesting to handle differences in times
It seems you are harvesting from an OAI-PMH v2.0 archive !
Use 86400 for a single day

Harvesting overlap [86400] :

[HARVESTING GRANULARITY]
This specifies whether harvesting is performed by day or second

It seems you are harvesting from an OAI-PMH v2.0 archive with only days!
Use 'day'
```

```
Harvesting granularity [day] :

[metadataPrefix]
This is the metadata format that will be harvested and stored.

Valid formats: oai_dc

metadataPrefix [] :
Please choose a value from the list above.
metadataPrefix [] : oai_dc

[set]
Choose a set to harvested (descriptions are in brackets).
Leave blank to harvest the whole archive.

Valid sets:

set [] :

Current list of archives:
0. naca.larc.nasa.gov http://naca.larc.nasa.gov/oai2.0/ oai_dc

Select from: [A]dd   [R]emove   [E]dit   [D]one
Enter your choice [D] : d

Finis.

Now you are ready to harvest
Run naca.larc.nasa.gov/harvest.pl to try harvesting with the default record
handler
```

4.1.2 populate-ntrs.pl

In particular, the “populate-ntrs.pl” file must be edited. The following snippet of code from that file provides an example of what should be added:

```
belvedere:/usr/local/web/htdocs/ntrs % less bin/populate-ntrs.pl
…
$archives{"biomedcentral.com"}{"id"} = "15";
$archives{"biomedcentral.com"}{"long_name"} = "BioMed Central";
$archives{"biomedcentral.com"}{"url"} = "http://www.biomedcentral.com/";
$archives{"biomedcentral.com"}{"nasa"} = "non-nasa";

$archives{"genesis.jpl.nasa.gov"}{"id"} = "16";
$archives{"genesis.jpl.nasa.gov"}{"long_name"} = "GENESIS (NASA Jet Propulsion
Laboratory)";
$archives{"genesis.jpl.nasa.gov"}{"url"} = "http://genesis2.jpl.nasa.gov/";
$archives{"genesis.jpl.nasa.gov"}{"nasa"} = "nasa";
…
```

New repositories should be assigned a unique integer identifier, given a user readable string that identifies them, a URL for the interface of that DL (this is not the same as the base URL used for OAI-PMH harvesting), and marked if they are of NASA origin or not.

At this point, the new repository will automatically be harvested by the Friday evening cron job (section 2.2.4). If you wish to harvest it now, you can cd run the “harvest.pl” script as indicated at the end of section 4.1.1

4.1.3 Generating New HTML Displays

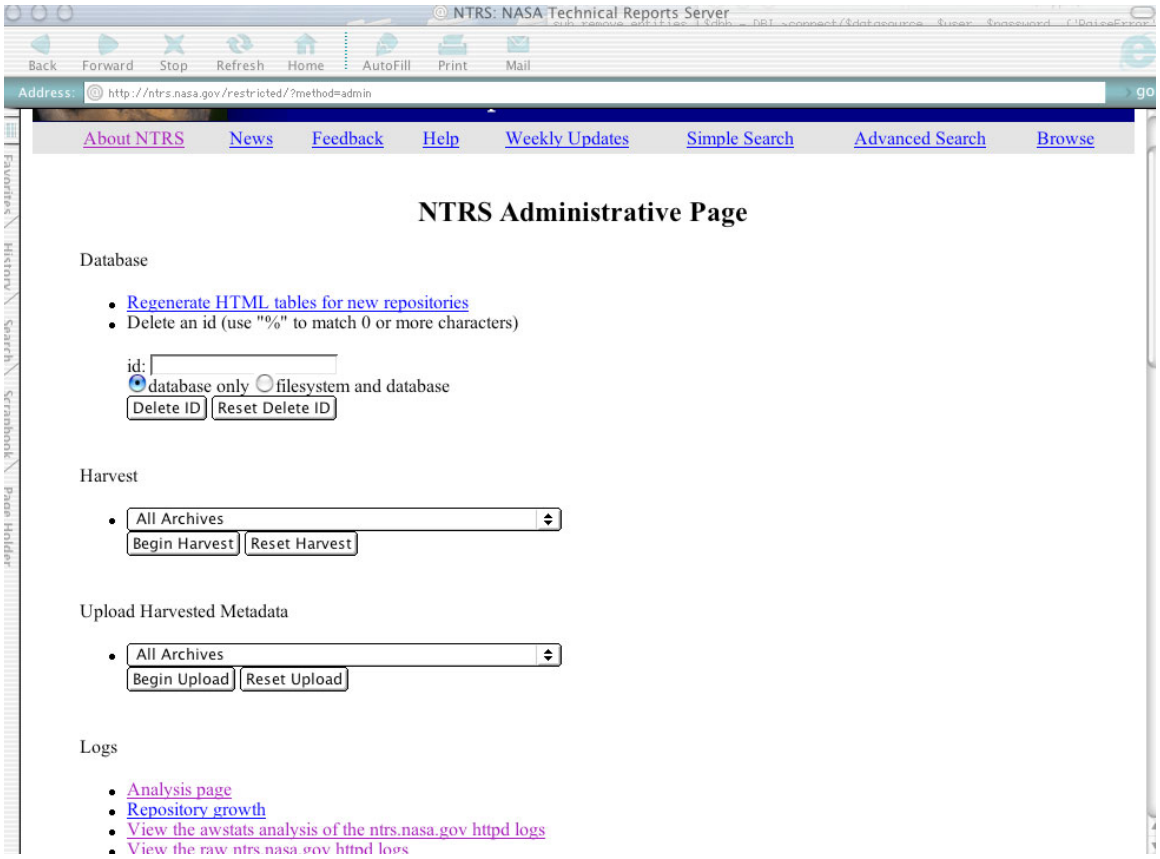
This last step is runnable from the command line or can be run from the administration interface (section 4.2). The purpose is to generate the HTML displays necessary for selecting from different repositories in either a HTML table with checkboxes (used by the advanced search interface), radio buttons (used by the browse interface), or as pull-down menu (also known as “select”, used by the updates and administration interface). The script is run in the following manner:

```
belvedere:/usr/local/web/htdocs/ntrs % bin/archives-html.pl
writing /usr/local/web/htdocs/ntrs/ntrs/ntrs.pkg/checkbox.html
writing /usr/local/web/htdocs/ntrs/ntrs/ntrs.pkg/radio.html
writing /usr/local/web/htdocs/ntrs/ntrs/ntrs.pkg/select.html
finished
```

4.2 Using the Administration Interface

The “admin” interface is shown in Figure 7. Because this interface is password protected and is not available to regular users, no direct link is provided; access is through the special URL:

http://ntrs.nasa.gov/?method=admin



http://ntrs.nasa.gov/restricted/?method=admin

About NTRS News Feedback Help Weekly Updates Simple Search Advanced Search Browse

NTRS Administrative Page

Database

- Regenerate HTML tables for new repositories
- Delete an id (use "%" to match 0 or more characters)

  id:
  [✓ database only] filesystem and database
  Delete ID | Reset Delete ID

Harvest

- All Archives
  Begin Harvest | Reset Harvest

Upload Harvested Metadata

- All Archives
  Begin Upload | Reset Upload

Logs

- Analysis page
- Repository growth
- View the awstats analysis of the ntrs.nasa.gov httpd logs
- View the raw ntrs.nasa.gov httpd logs

Figure 7. The Admin Interface

There are more capabilities than are shown in Figure 7. The first link, “Regenerate HTML Tables”, is the same as running the “bin/archives-html.pl” script as described in section 4.1.3. The deletion capability can be used to delete an ID from the database or from the database and the filesystem.

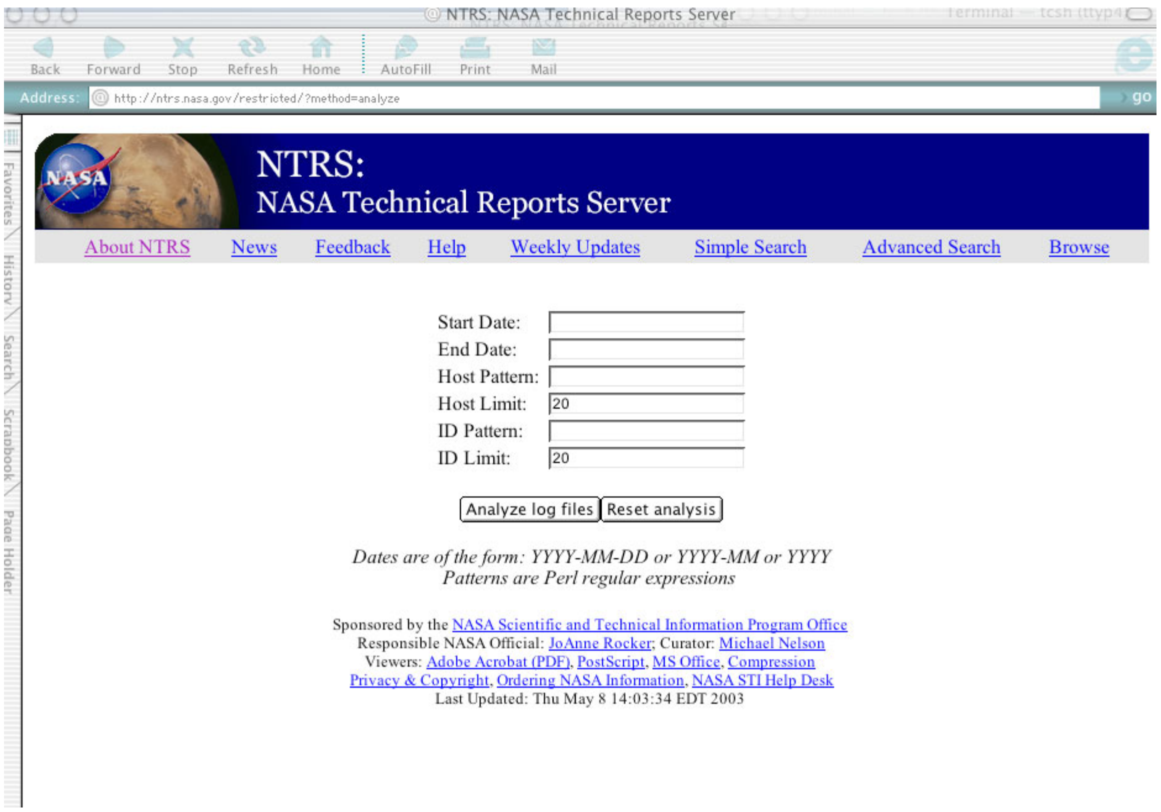
The “Harvest” and “Upload Harvested Metadata” sections allow the same functionality as running the “bin/harvest-cron” and “bin/populate-ntrs.pl” scripts by hand, respectively. The “Logs” portion is covered in section 4.3

There is also links for viewing the collected feedback (contributed by users via the “feedback” method) in raw or processed form.

There are also user-friend interface for standard bucket methods, such as Terms & Condition management, principal management, and bucket integrity checking. There is also a form to retrieve and upload new RSS files for the “news” method. Except for the actions covered in section 4.1, all NTRS management functions should be available through the admin interface.

4.3 Analyzing Usage

All of the raw bucket logs and apache http logs are available for perusal from this section of the administration interface. However, the interesting capabilities are in the analysis functionalities provided. Figure 8 shows the analysis interface, in which the top documents and clients can be determined for a given time period. Figure 9 shows the interface to track the growth of the holdings on a per-repository basis over time. Figure 10 shows part of the detailed log analysis that awstats provides.

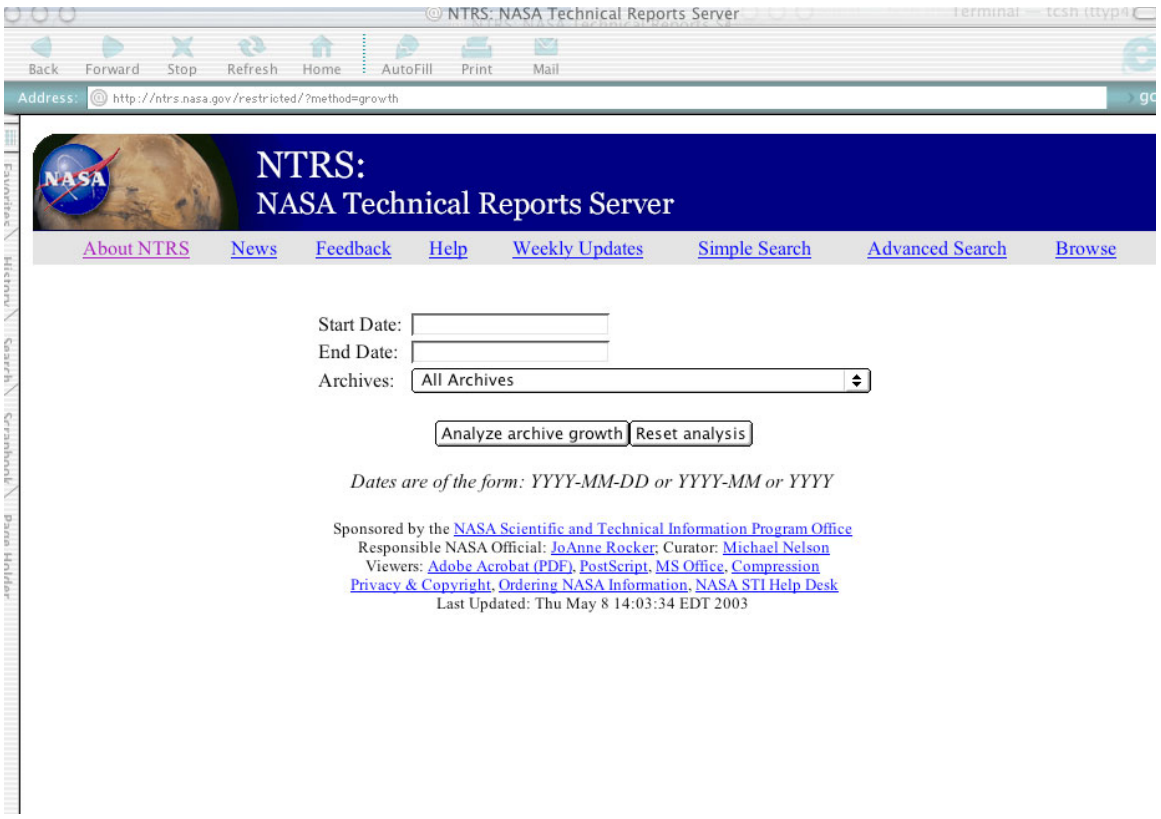


Start Date:	
End Date:	
Host Pattern:	
Host Limit:	20
ID Pattern:	
ID Limit:	20

Dates are of the form: YYYY-MM-DD or YYYY-MM or YYYY
Patterns are Perl regular expressions

Sponsored by the NASA Scientific and Technical Information Program Office
Responsible NASA Official: JoAnne Rocker; Curator: Michael Nelson
Viewers: Adobe Acrobat (PDF), PostScript, MS Office, Compression
Privacy & Copyright, Ordering NASA Information, NASA STI Help Desk
Last Updated: Thu May 8 14:03:34 EDT 2003

Figure 8. Analysis Interface

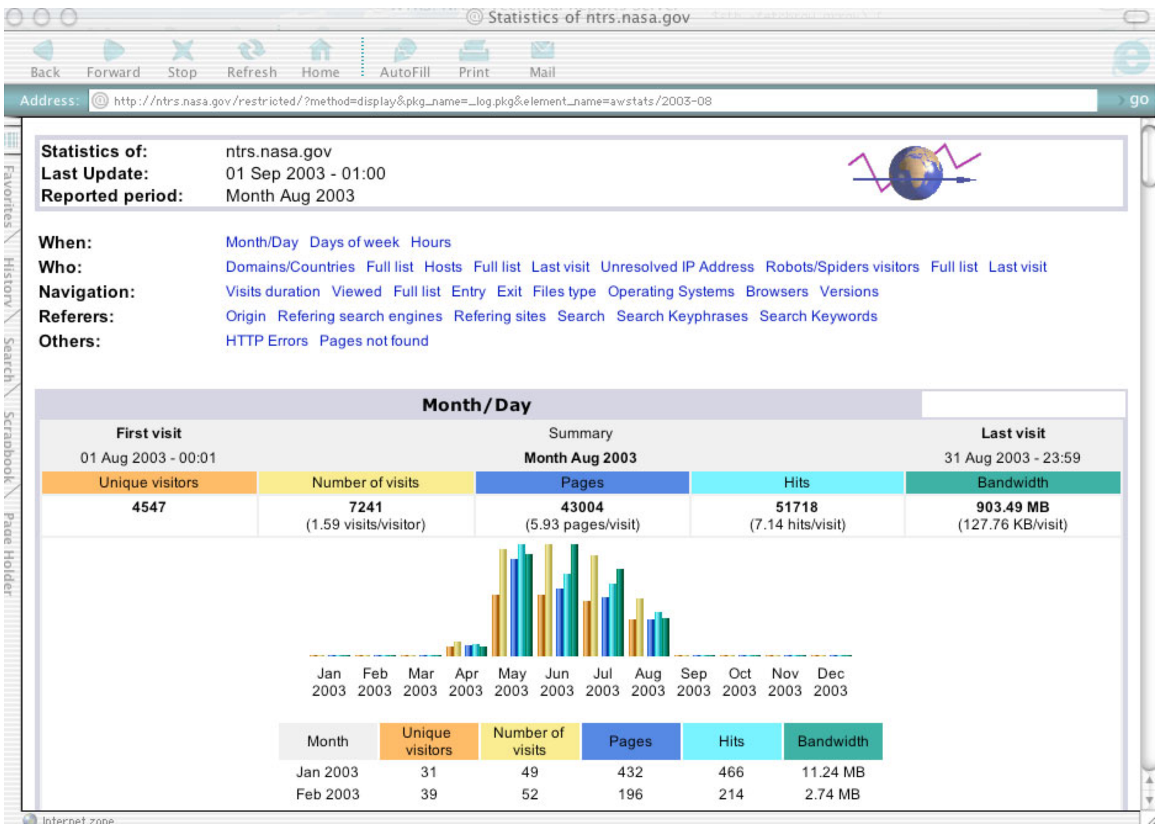


Start Date:	
End Date:	
Archives:	All Archives

Dates are of the form: YYYY-MM-DD or YYYY-MM or YYYY

Sponsored by the NASA Scientific and Technical Information Program Office
Responsible NASA Official: JoAnne Rocker; Curator: Michael Nelson
Viewers: Adobe Acrobat (PDF), PostScript, MS Office, Compression
Privacy & Copyright, Ordering NASA Information, NASA STI Help Desk
Last Updated: Thu May 8 14:03:34 EDT 2003

Figure 9. Growth Interface.



http://ntrs.nasa.gov/restricted/?method=display&pkg_name=_log.pkg&element_name=awstats/2003-08

Statistics of:	ntrs.nasa.gov
Last Update:	01 Sep 2003 - 01:00
Reported period:	Month Aug 2003

When:	Month/Day Days of week Hours
Who:	Domains/Countries Full list Hosts Full list Last visit Unresolved IP Address Robots/Spiders visitors Full list Last visit
Navigation:	Visits duration Viewed Full list Entry Exit Files type Operating Systems Browsers Versions
Referers:	Origin Refering search engines Refering sites Search Search Keyphrases Search Keywords
Others:	HTTP Errors Pages not found

Month/Day

First visit		Summary		Last visit
01 Aug 2003 - 00:01		Month Aug 2003		31 Aug 2003 - 23:59
Unique visitors	Number of visits	Pages	Hits	Bandwidth
4547	7241 (1.59 visits/visitor)	43004 (5.93 pages/visit)	51718 (7.14 hits/visit)	903.49 MB (127.76 KB/visit)

Month	Unique visitors	Number of visits	Pages	Hits	Bandwidth
Jan 2003	31	49	432	466	11.24 MB
Feb 2003	39	52	196	214	2.74 MB

Figure 10. Awstats Display.

5. Conclusions and Future Work

This report documents the design and operation of the NTRS digital library. NTRS is implemented as a specialized bucket, and uses a variety of technologies to provide its services, including an OAI-PMH harvester, an OAI-PMH repository (thus making NTRS an aggregator), a MySQL database, the awstats http log analysis facility, and a variety of support scripts to integrate the various aspects.

This model for implementing DLs appears to be very successful. We have been able to replicate this model for a version of the NACA Technical Report Server, currently under testing by NASA staff. The operation, support scripts and related technologies were adapted to this new DL in a short amount of time. A separate report about the NACA Technical Report Server will be issued when it is fully tested and accepted by NASA staff members.

User feedback, staff comments and NTRS usage analysis have uncovered a variety of areas for future work on NTRS:

- increased metadata, especially for the NASA CASI records. The current NTRS does not display the accession identifiers, which are used by many libraries to locate the reports

- increased holdings. The distributed searching version of NTRS included access to metadata from the AIAA, IEEE and other publishers and professional societies. Users have requested that this material be returned, although the intellectual property rights of this metadata are still unclear at the moment.

- increased NASA participation. We anticipate adding NASA JPL to NTRS in the near future, with NASA GRC hopefully to follow soon. NASA DFRC has a significant DL that could be easily made OAI-PMH compliant, but we are unsure of their status. NASA ARC, GSFC, SSC and KSC do not appear, at the moment, to be interested in running their own DL at all. Their contents will have to come through CASI participation. NASA CASI participation is being worked on under a separate project between NASA and Old Dominion University; its status is beyond the scope of this report.

- increased non-NASA participation. We already have four high-quality non-NASA repositories, but we would like to add more. A variety of targets have been discussed in the short and medium-range, including: all aerospace related theses and dissertations available from OCLC; the public holdings of the Naval Research Laboratory; and the public holdings of Los Alamos National Laboratory. We will continue to review other appropriate repositories as they come on-line.

6. References

Kahle, B., Morris, H., Davis, F., Tienne, K., Hart, C. & Palmer, R. (1992). Wide Area Information Servers: An Executive Information System for Unstructured Files. Electronic Networking: Research, Applications and Policy, 2(1), 59-68.

Kurtz, M. J., Eichorn, G., Accomazzi, A., Grant, C. S., Demleitner, M. & Murray, S. S. (1999). The NASA ADS abstract service and the distributed astronomy digital library. D-Lib Magazine, 5(11). Available at http://www.dlib.org/dlib/november99/11kurtz.html.

Lagoze, C., Van de Sompel, H., Nelson, M. L. & Warner, S. (2002). The Open Archives Initiative Protocol for Metadata Harvesting. Available at: http://www.openarchives.org/OAI/openarchivesprotocol.html

Liu, X., Maly, K., Zubair, M. & Nelson, M. L. (2001) Arc: an OAI service provider for digital library federation. D-Lib Magazine, 7(4). Available at http://www.dlib.org/dlib/april01/liu/04liu.html

Lynch, C. A. (1997). The Z39.50 Information Retrieval Standard. D-Lib Magazine, 3(4). Available at http://www.dlib.org/dlib/april97/04lynch.html

Maly, K., Nelson, M. L., & Zubair, M. (1999). Smart Objects, Dumb Archives: A User-Centric, Layered Digital Library Framework. D-Lib Magazine, 5(3). Available at: http://www.dlib.org/dlib/march99/maly/03maly.html

Nelson, M. L., Rocker, J. & Harrison, T. L. (2003). OAI and NASA Scientific and Technical Information, Library Hi-Tech, 21(2).

Nelson, M. L., Gottlich, G. L., & Bianco, D. J. (1994). World Wide Web implementation of the Langley technical report server. NASA TM-109162. Available at http://techreports.larc.nasa.gov/ltrs/PDF/tm109162.pdf

Nelson, M. L., Gottlich, G. L., Bianco, D. J., Paulson, S. S., Binkley, R. L., Kellogg, Y. D., Beaumont, C. J., Schmunk, R. B., Kurtz, M. J., Accomazzi, A., & Syed, O. (1995). The NASA technical report server. Internet Research: Electronic Network Applications and Policy, 5(2), 25-36. Available at: http://techreports.larc.nasa.gov/ltrs/papers/NASA-95-ir-p25/NASA-95-ir-p25.html

Nelson, M. L. (2000). Buckets: Smart Objects for Digital Libraries. PhD Dissertation, Computer Science Department, Old Dominion University. Available at: http://www.cs.odu.edu/~mln/phd/

Powell, A. L. & French, J. C. (2000) Growth and server availability for the NCSTRL digital library. Proceedings of the 5th ACM Conference on Digital Libraries, pp. 264-265.

Roper, D. G., McCaskill, M. K., Holland, S. D., Walsh, J. L., Nelson, M. L., Adkins, S. L., Ambur, M. Y., & Campbell, B. A. (1994). A strategy for electronic dissemination of NASA Langley publications. NASA TM-109172. Available at http://techreports.larc.nasa.gov/ltrs/PDF/tm109172.pdf

Sidwell, C. A., Needham, P. A. S. & Harrington, J. D. (2000). Lightening grey literature: Making the invisible visible. New Review of Information Networking, 6, 121-136.

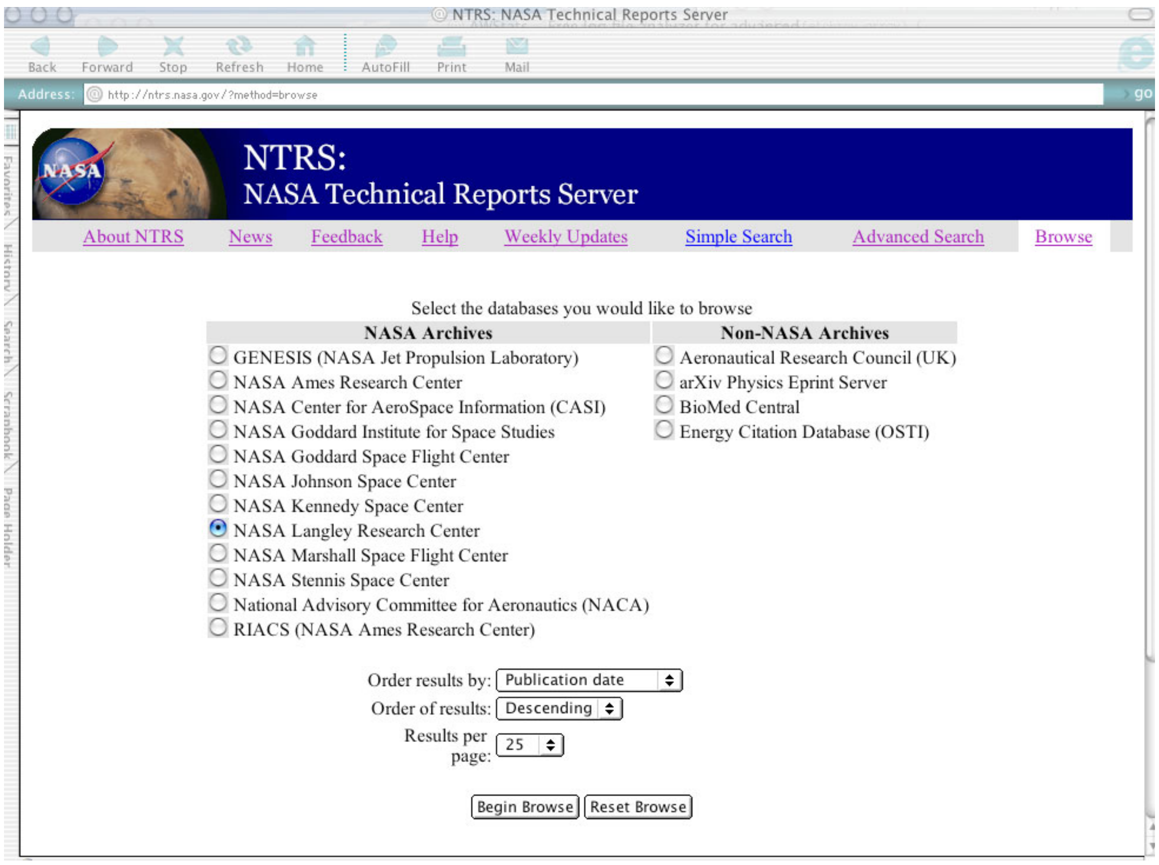
Suleman, H. & Fox, E. A. (2001). A framework for building open digital libraries. D-Lib Magazine, 7(12). Available at http://www.dlib.org/dlib/december01/suleman/12suleman.html

Van de Sompel, H., Krichel, T., Nelson, M. L., Hochstenbach, P., Lyapunov, V. M., Maly, K., Zubair, M., Kholief, M., Liu, X. & O' Connell, H. (2000). The UPS prototype: an experimental end-user service across e-print archives. D-Lib Magazine, 6(2). Available at http://www.dlib.org/dlib/february00/vandesompel-ups/02vandesompel-ups.html

Weibel, S. & Lagoze, C. (1997). An Element Set to Support Resource Discovery - The State of the Dublin Core. International Journal on Digital Libraries. 1(2), 176-186.

Appendix 1: HTML Output of NTRS-Specific Bucket Methods

The “simple”, “advanced”, “admin”, “analyze” and “growth” methods have already been shown in Figure 4, 5, 7, 8, and 9, respectively.

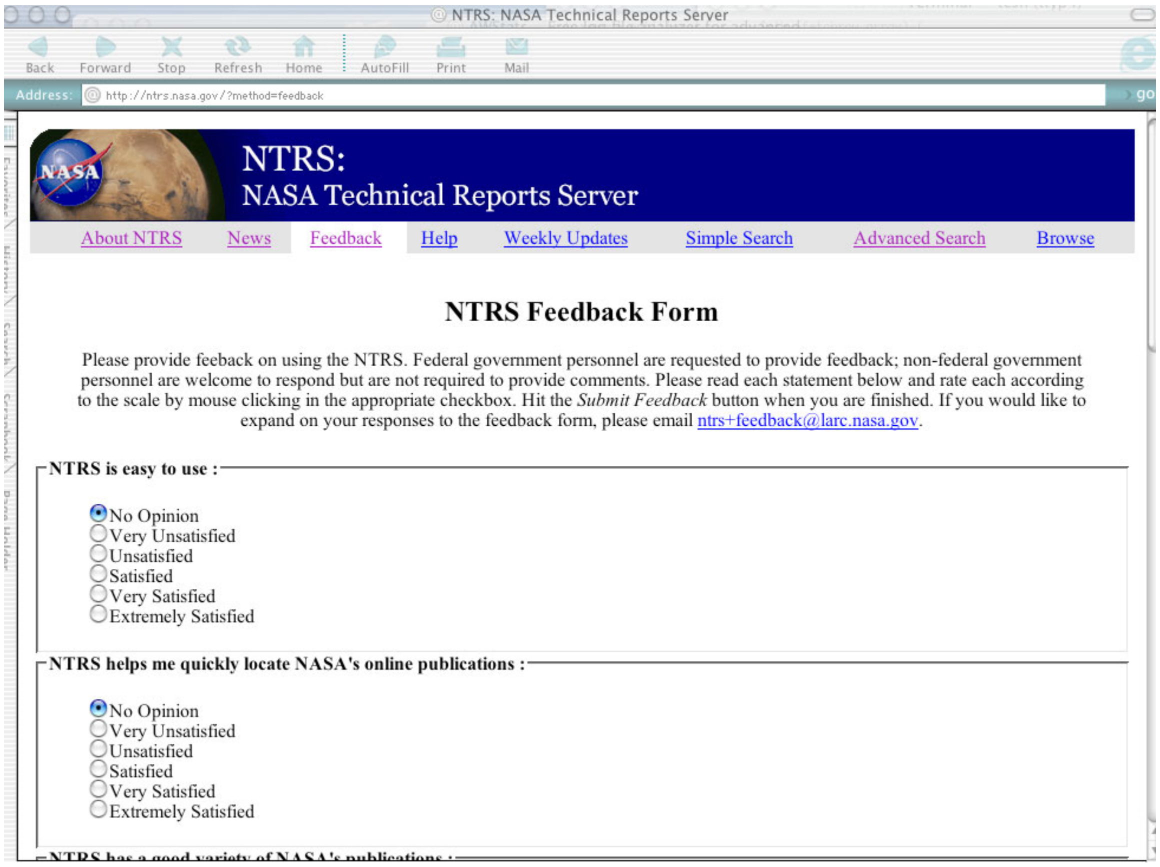


Select the databases you would like to browse

NASA Archives	Non-NASA Archives
○ GENESIS (NASA Jet Propulsion Laboratory)	○ Aeronautical Research Council (UK)
○ NASA Ames Research Center	○ arXiv Physics Eprint Server
○ NASA Center for AeroSpace Information (CASI)	○ BioMed Central
○ NASA Goddard Institute for Space Studies	○ Energy Citation Database (OSTI)
○ NASA Goddard Space Flight Center	
○ NASA Johnson Space Center	
○ NASA Kennedy Space Center	
[✓ NASA Langley Research Center]	
○ NASA Marshall Space Flight Center	
○ NASA Stennis Space Center	
○ National Advisory Committee for Aeronautics (NACA)	
○ RIACS (NASA Ames Research Center)	

Order results by: Publication date
Order of results: Descending
Results per page: 25

Figure A1.1. The Browse Interface



NTRS Feedback Form

Please provide feeback on using the NTRS. Federal government personnel are requested to provide feedback; non-federal government personnel are welcome to respond but are not required to provide comments. Please read each statement below and rate each according to the scale by mouse clicking in the appropriate checkbox. Hit the *Submit Feedback* button when you are finished. If you would like to expand on your responses to the feedback form, please email ntrs+feedback@larc.nasa.gov.

NTRS is easy to use :

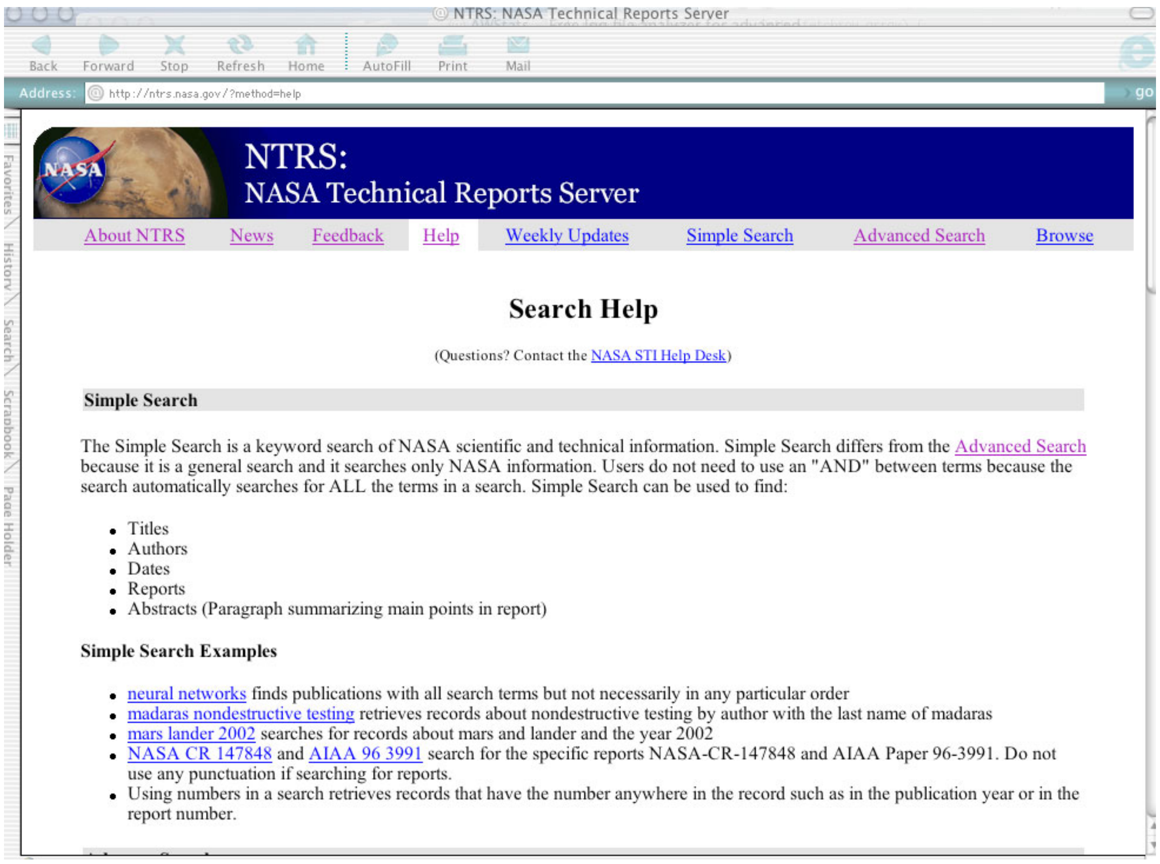
[✓ No Opinion]
Very Unsatisfied
Unsatisfied
Satisfied
Very Satisfied
Extremely Satisfied

NTRS helps me quickly locate NASA's online publications :

[✓ No Opinion]
Very Unsatisfied
Unsatisfied
Satisfied
Very Satisfied
Extremely Satisfied

NTRS has a good variety of NASA's publications :

Figure A1.2. The Feedback Interface.



NTRS:
NASA Technical Reports Server

About NTRS | News | Feedback | Help | Weekly Updates | Simple Search | Advanced Search | Browse

Search Help

(Questions? Contact the NASA STI Help Desk)

Simple Search

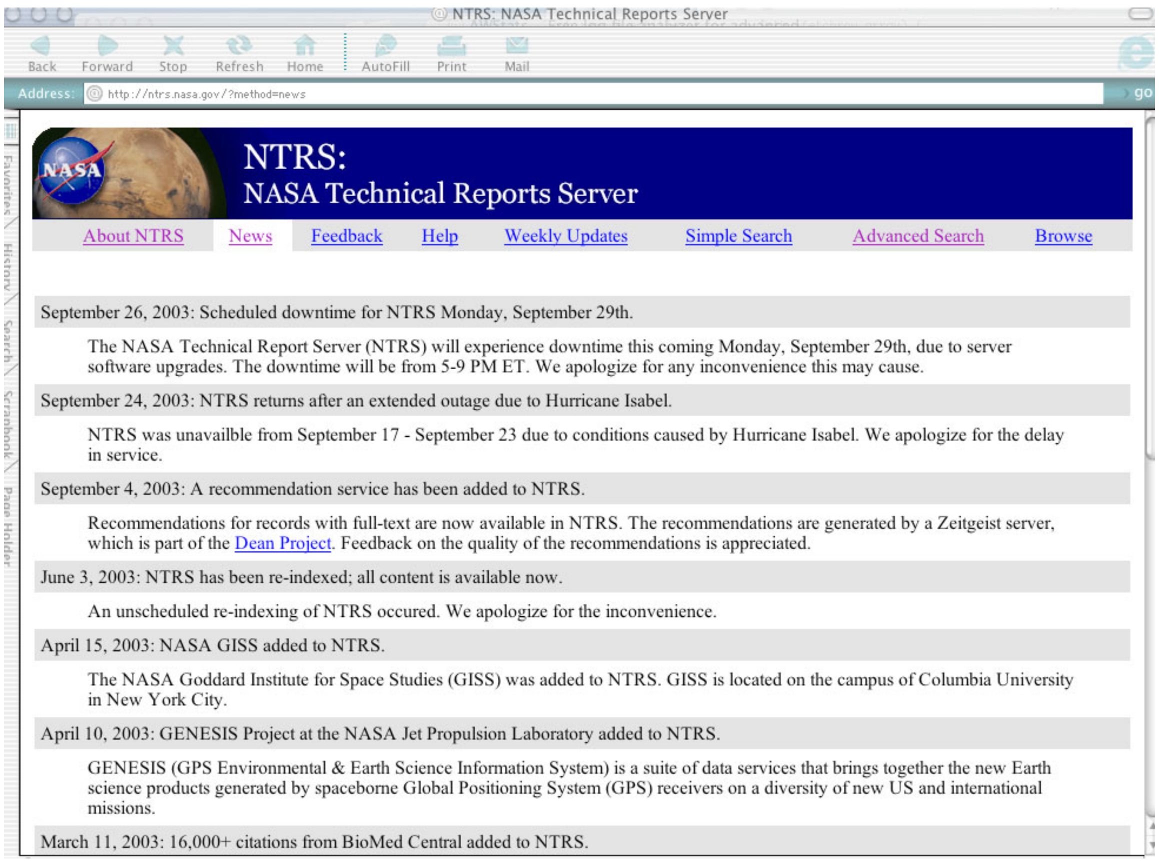
The Simple Search is a keyword search of NASA scientific and technical information. Simple Search differs from the Advanced Search because it is a general search and it searches only NASA information. Users do not need to use an "AND" between terms because the search automatically searches for ALL the terms in a search. Simple Search can be used to find:

- Titles
- Authors
- Dates
- Reports
- Abstracts (Paragraph summarizing main points in report)

Simple Search Examples

- neural networks finds publications with all search terms but not necessarily in any particular order
- madaras nondestructive testing retrieves records about nondestructive testing by author with the last name of madaras
- mars lander 2002 searches for records about mars and lander and the year 2002
- NASA CR 147848 and AIAA 96 3991 search for the specific reports NASA-CR-147848 and AIAA Paper 96-3991. Do not use any punctuation if searching for reports.
- Using numbers in a search retrieves records that have the number anywhere in the record such as in the publication year or in the report number.

Figure A1.3. The Context-Sensitive Help Interface.



NTRS:
NASA Technical Reports Server

About NTRS | News | Feedback | Help | Weekly Updates | Simple Search | Advanced Search | Browse

September 26, 2003: Scheduled downtime for NTRS Monday, September 29th.

The NASA Technical Report Server (NTRS) will experience downtime this coming Monday, September 29th, due to server software upgrades. The downtime will be from 5-9 PM ET. We apologize for any inconvenience this may cause.

September 24, 2003: NTRS returns after an extended outage due to Hurricane Isabel.

NTRS was unavailble from September 17 - September 23 due to conditions caused by Hurricane Isabel. We apologize for the delay in service.

September 4, 2003: A recommendation service has been added to NTRS.

Recommendations for records with full-text are now available in NTRS. The recommendations are generated by a Zeitgeist server, which is part of the Dean Project. Feedback on the quality of the recommendations is appreciated.

June 3, 2003: NTRS has been re-indexed; all content is available now.

An unscheduled re-indexing of NTRS occured. We apologize for the inconvenience.

April 15, 2003: NASA GISS added to NTRS.

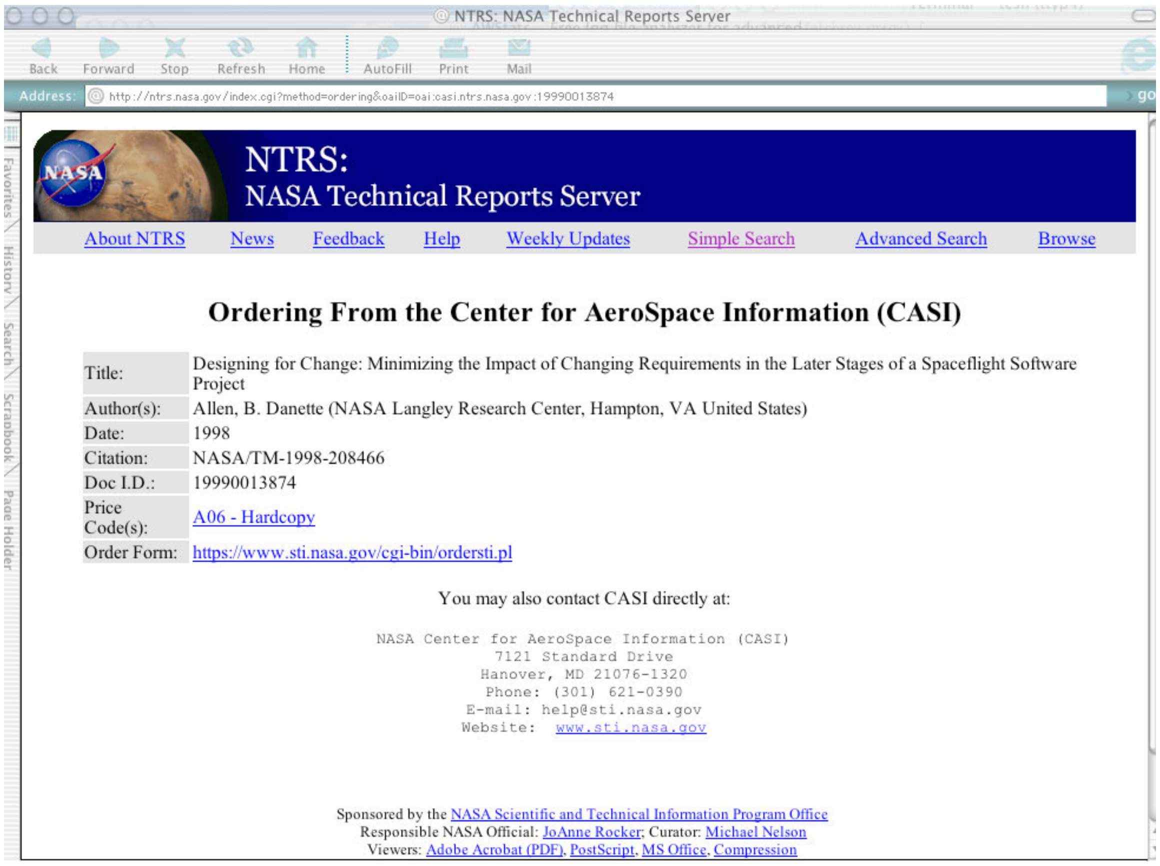
The NASA Goddard Institute for Space Studies (GISS) was added to NTRS. GISS is located on the campus of Columbia University in New York City.

April 10, 2003: GENESIS Project at the NASA Jet Propulsion Laboratory added to NTRS.

GENESIS (GPS Environmental & Earth Science Information System) is a suite of data services that brings together the new Earth science products generated by spaceborne Global Positioning System (GPS) receivers on a diversity of new US and international missions.

March 11, 2003: 16,000+ citations from BioMed Central added to NTRS.

Figure A1.4. The New Interface.



Ordering From the Center for AeroSpace Information (CASI)

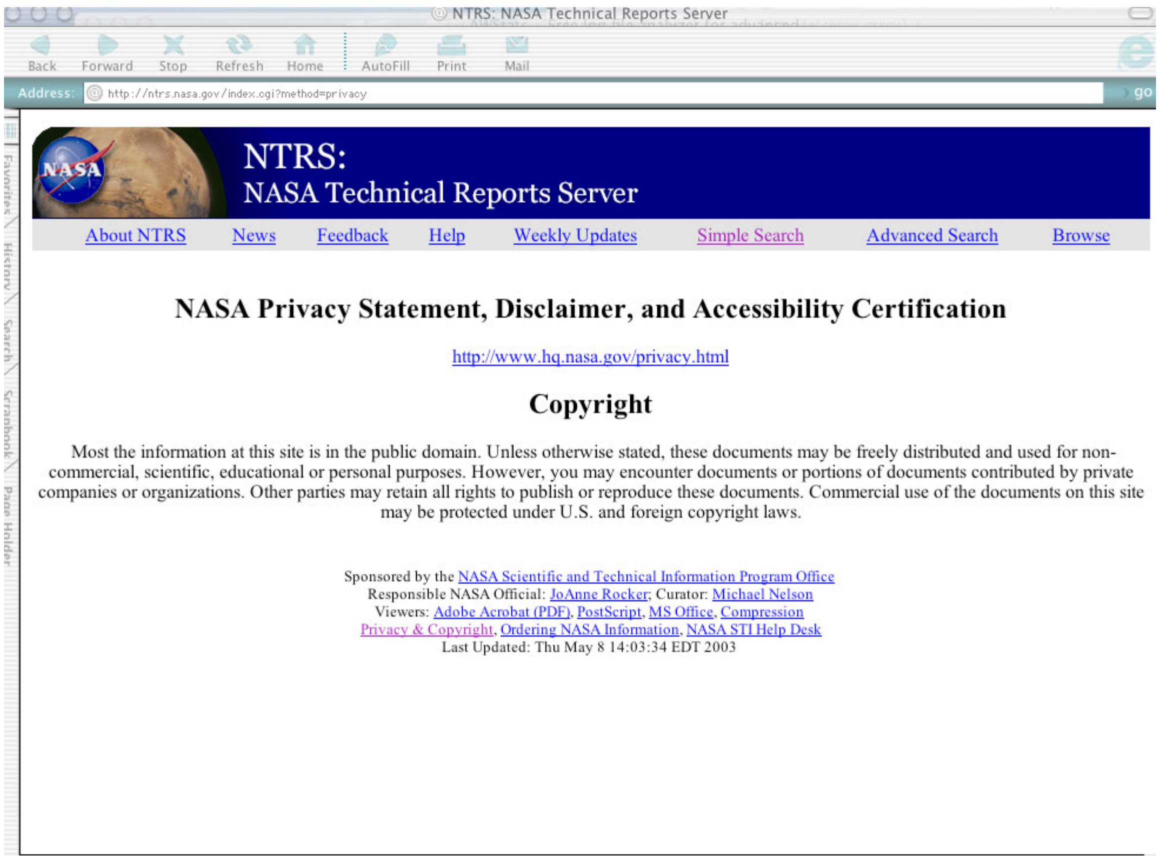
Title:	Designing for Change: Minimizing the Impact of Changing Requirements in the Later Stages of a Spaceflight Software Project
Author(s):	Allen, B. Danette (NASA Langley Research Center, Hampton, VA United States)
Date:	1998
Citation:	NASA/TM-1998-208466
Doc I.D.:	19990013874
Price Code(s):	A06 - Hardcopy
Order Form:	https://www.sti.nasa.gov/cgi-bin/ordersti.pl

You may also contact CASI directly at:

```
NASA Center for AeroSpace Information (CASI)
          7121 Standard Drive
        Hanover, MD 21076-1320
         Phone: (301) 621-0390
       E-mail: help@sti.nasa.gov
      Website:  www.sti.nasa.gov
```

Sponsored by the NASA Scientific and Technical Information Program Office
Responsible NASA Official: JoAnne Rocker; Curator: Michael Nelson
Viewers: Adobe Acrobat (PDF), PostScript, MS Office, Compression

Figure A1.5. The Ordering Interface.



http://ntrs.nasa.gov/index.cgi?method=privacy

NTRS:
NASA Technical Reports Server

About NTRS | News | Feedback | Help | Weekly Updates | Simple Search | Advanced Search | Browse

NASA Privacy Statement, Disclaimer, and Accessibility Certification

http://www.hq.nasa.gov/privacy.html

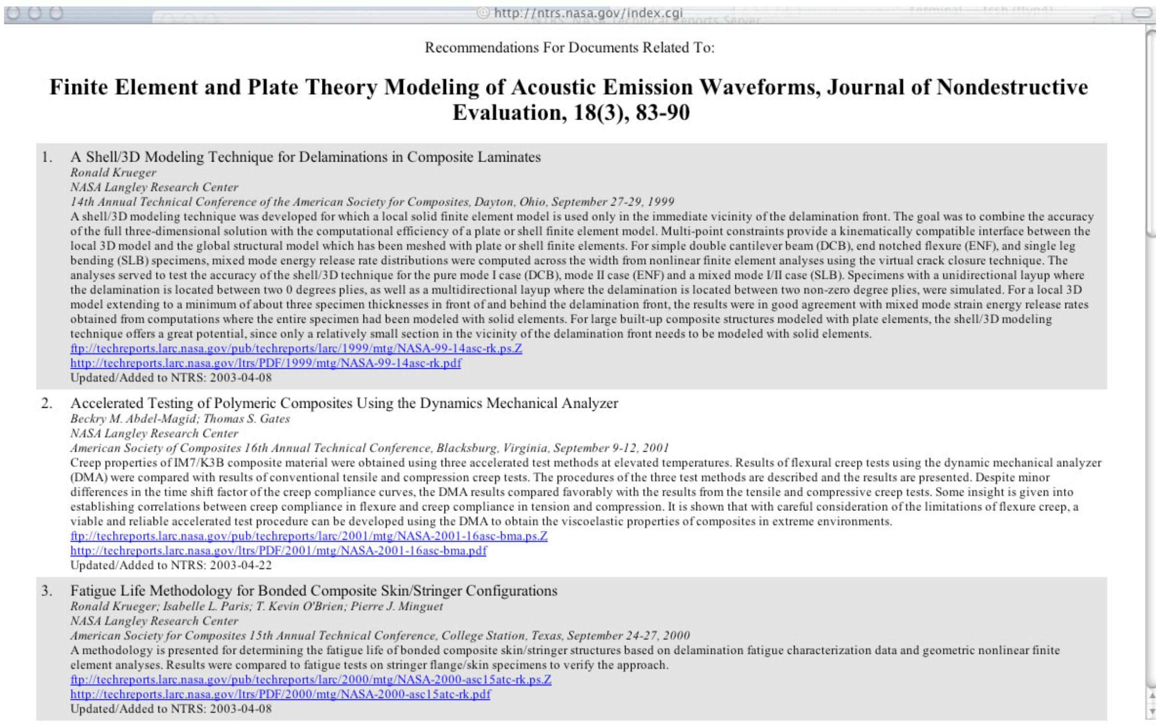
Copyright

Most the information at this site is in the public domain. Unless otherwise stated, these documents may be freely distributed and used for non-commercial, scientific, educational or personal purposes. However, you may encounter documents or portions of documents contributed by private companies or organizations. Other parties may retain all rights to publish or reproduce these documents. Commercial use of the documents on this site may be protected under U.S. and foreign copyright laws.

Sponsored by the NASA Scientific and Technical Information Program Office
Responsible NASA Official: JoAnne Rocker; Curator: Michael Nelson
Viewers: Adobe Acrobat (PDF), PostScript, MS Office, Compression
Privacy & Copyright, Ordering NASA Information, NASA STI Help Desk
Last Updated: Thu May 8 14:03:34 EDT 2003

Figure A1.6. The Privacy Statement

.

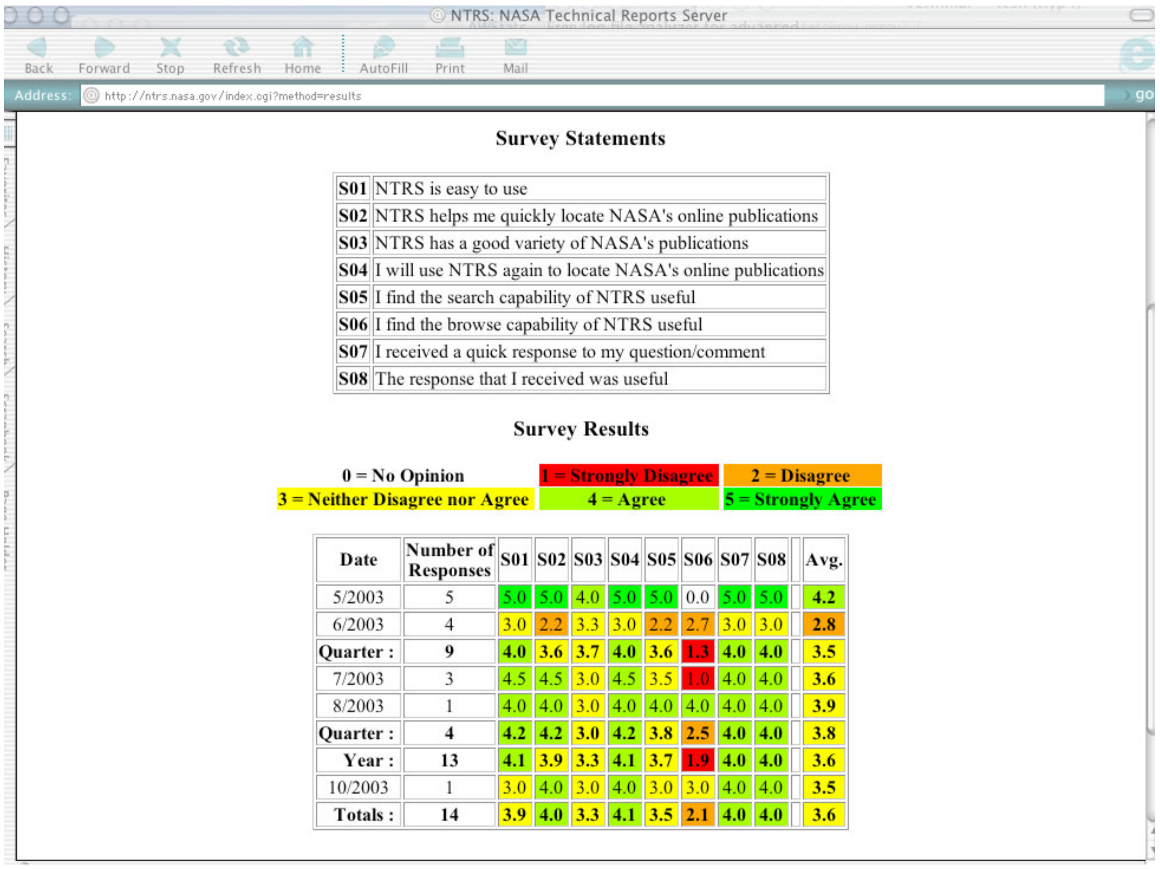


Recommendations For Documents Related To:

Finite Element and Plate Theory Modeling of Acoustic Emission Waveforms, Journal of Nondestructive Evaluation, 18(3), 83-90

1. A Shell/3D Modeling Technique for Delaminations in Composite Laminates
   Ronald Krueger
   NASA Langley Research Center
   14th Annual Technical Conference of the American Society for Composites, Dayton, Ohio, September 27-29, 1999
   A shell/3D modeling technique was developed for which a local solid finite element model is used only in the immediate vicinity of the delamination front. The goal was to combine the accuracy of the full three-dimensional solution with the computational efficiency of a plate or shell finite element model. Multi-point constraints provide a kinematically compatible interface between the local 3D model and the global structural model which has been meshed with plate or shell finite elements. For simple double cantilever beam (DCB), end notched flexure (ENF), and single leg bending (SLB) specimens, mixed mode energy release rate distributions were computed across the width from nonlinear finite element analyses using the virtual crack closure technique. The analyses served to test the accuracy of the shell/3D technique for the pure mode I case (DCB), mode II case (ENF) and a mixed mode I/II case (SLB). Specimens with a unidirectional layup where the delamination is located between two 0 degrees plies, as well as a multidirectional layup where the delamination is located between two non-zero degree plies, were simulated. For a local 3D model extending to a minimum of about three specimen thicknesses in front of and behind the delamination front, the results were in good agreement with mixed mode strain energy release rates obtained from computations where the entire specimen had been modeled with solid elements. For large built-up composite structures modeled with plate elements, the shell/3D modeling technique offers a great potential, since only a relatively small section in the vicinity of the delamination front needs to be modeled with solid elements.
   ftp://techreports.larc.nasa.gov/pub/techreports/larc/1999/mtg/NASA-99-14asc-rk.ps.Z
   http://techreports.larc.nasa.gov/ltrs/PDF/1999/mtg/NASA-99-14asc-rk.pdf
   Updated/Added to NTRS: 2003-04-08
2. Accelerated Testing of Polymeric Composites Using the Dynamics Mechanical Analyzer
   Beckry M. Abdel-Magid; Thomas S. Gates
   NASA Langley Research Center
   American Society of Composites 16th Annual Technical Conference, Blacksburg, Virginia, September 9-12, 2001
   Creep properties of IM7/K3B composite material were obtained using three accelerated test methods at elevated temperatures. Results of flexural creep tests using the dynamic mechanical analyzer (DMA) were compared with results of conventional tensile and compression creep tests. The procedures of the three test methods are described and the results are presented. Despite minor differences in the time shift factor of the creep compliance curves, the DMA results compared favorably with the results from the tensile and compressive creep tests. Some insight is given into establishing correlations between creep compliance in flexure and creep compliance in tension and compression. It is shown that with careful consideration of the limitations of flexure creep, a viable and reliable accelerated test procedure can be developed using the DMA to obtain the viscoelastic properties of composites in extreme environments.
   ftp://techreports.larc.nasa.gov/pub/techreports/larc/2001/mtg/NASA-2001-16asc-bma.ps.Z
   http://techreports.larc.nasa.gov/ltrs/PDF/2001/mtg/NASA-2001-16asc-bma.pdf
   Updated/Added to NTRS: 2003-04-22
3. Fatigue Life Methodology for Bonded Composite Skin/Stringer Configurations
   Ronald Krueger; Isabelle L. Paris; T. Kevin O'Brien; Pierre J. Minguet
   NASA Langley Research Center
   American Society for Composites 15th Annual Technical Conference, College Station, Texas, September 24-27, 2000
   A methodology is presented for determining the fatigue life of bonded composite skin/stringer structures based on delamination fatigue characterization data and geometric nonlinear finite element analyses. Results were compared to fatigue tests on stringer flange/skin specimens to verify the approach.
   ftp://techreports.larc.nasa.gov/pub/techreports/larc/2000/mtg/NASA-2000-asc15atc-rk.ps.Z
   http://techreports.larc.nasa.gov/ltrs/PDF/2000/mtg/NASA-2000-asc15atc-rk.pdf
   Updated/Added to NTRS: 2003-04-08

Figure A1.7. Recommendations Generated for A Document.



NTRS: NASA Technical Reports Server

Survey Statements

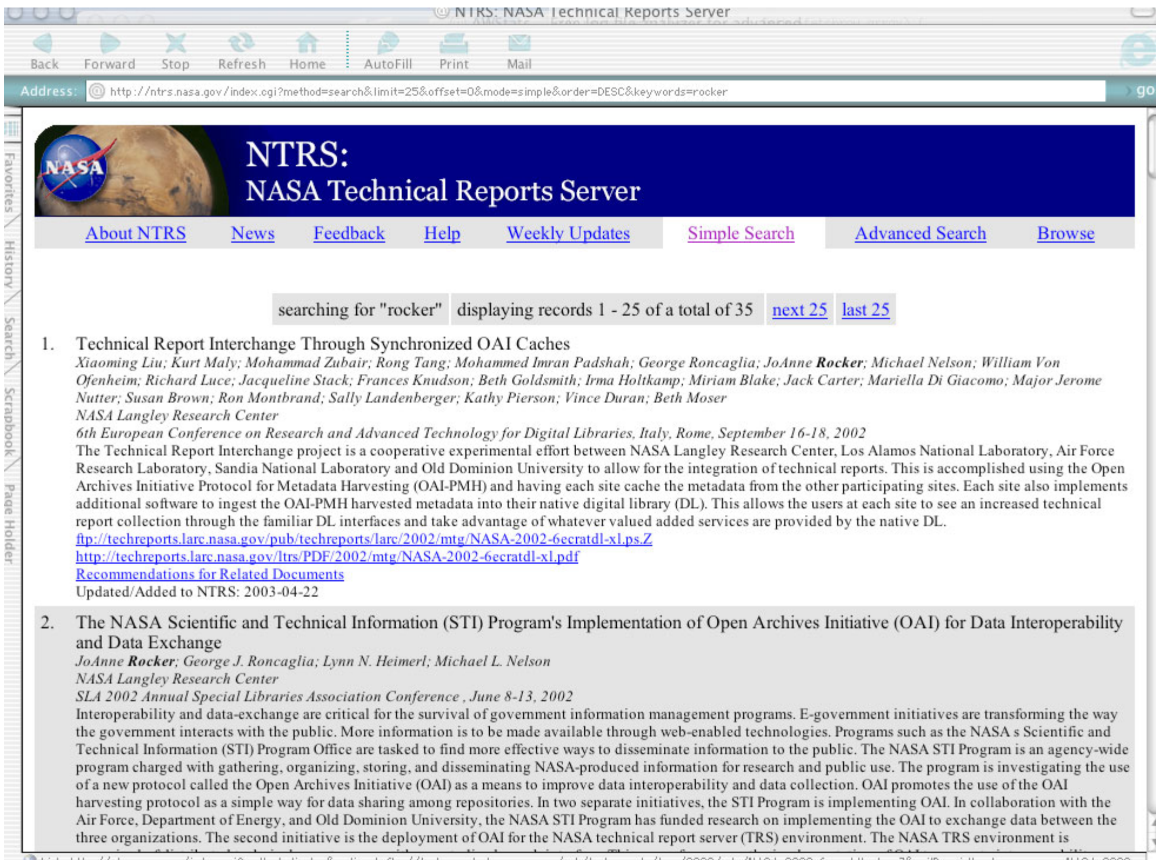
S01	NTRS is easy to use
S02	NTRS helps me quickly locate NASA's online publications
S03	NTRS has a good variety of NASA's publications
S04	I will use NTRS again to locate NASA's online publications
S05	I find the search capability of NTRS useful
S06	I find the browse capability of NTRS useful
S07	I received a quick response to my question/comment
S08	The response that I received was useful

Survey Results

0 = No Opinion	**1 = Strongly Disagree**	**2 = Disagree**
3 = Neither Disagree nor Agree	**4 = Agree**	**5 = Strongly Agree**

Date	Number of Responses	S01	S02	S03	S04	S05	S06	S07	S08		Avg.
5/2003	5	5.0	5.0	4.0	5.0	5.0	0.0	5.0	5.0		**4.2**
6/2003	4	3.0	2.2	3.3	3.0	2.2	2.7	3.0	3.0		**2.8**
Quarter :	**9**	**4.0**	**3.6**	**3.7**	**4.0**	**3.6**	**1.3**	**4.0**	**4.0**		**3.5**
7/2003	3	4.5	4.5	3.0	4.5	3.5	1.0	4.0	4.0		**3.6**
8/2003	1	4.0	4.0	3.0	4.0	4.0	4.0	4.0	4.0		**3.9**
Quarter :	**4**	**4.2**	**4.2**	**3.0**	**4.2**	**3.8**	**2.5**	**4.0**	**4.0**		**3.8**
Year :	**13**	**4.1**	**3.9**	**3.3**	**4.1**	**3.7**	**1.9**	**4.0**	**4.0**		**3.6**
10/2003	1	3.0	4.0	3.0	4.0	3.0	3.0	4.0	4.0		**3.5**
Totals :	**14**	**3.9**	**4.0**	**3.3**	**4.1**	**3.5**	**2.1**	**4.0**	**4.0**		**3.6**

Figure A1.8. Summarized User Feedback Results.


NTRS: NASA Technical Reports Server
Back
Forward
Stop
Refresh
Home
AutoFill
Print
Mail
Address:
http://ntrs.nasa.gov/index.cgi?method=search&limit=25&offset=0&mode=simple&order=DESC&keywords=rocker
go
Favorites
History
Search
Scrapbook
Page Holder

NTRS:
NASA Technical Reports Server

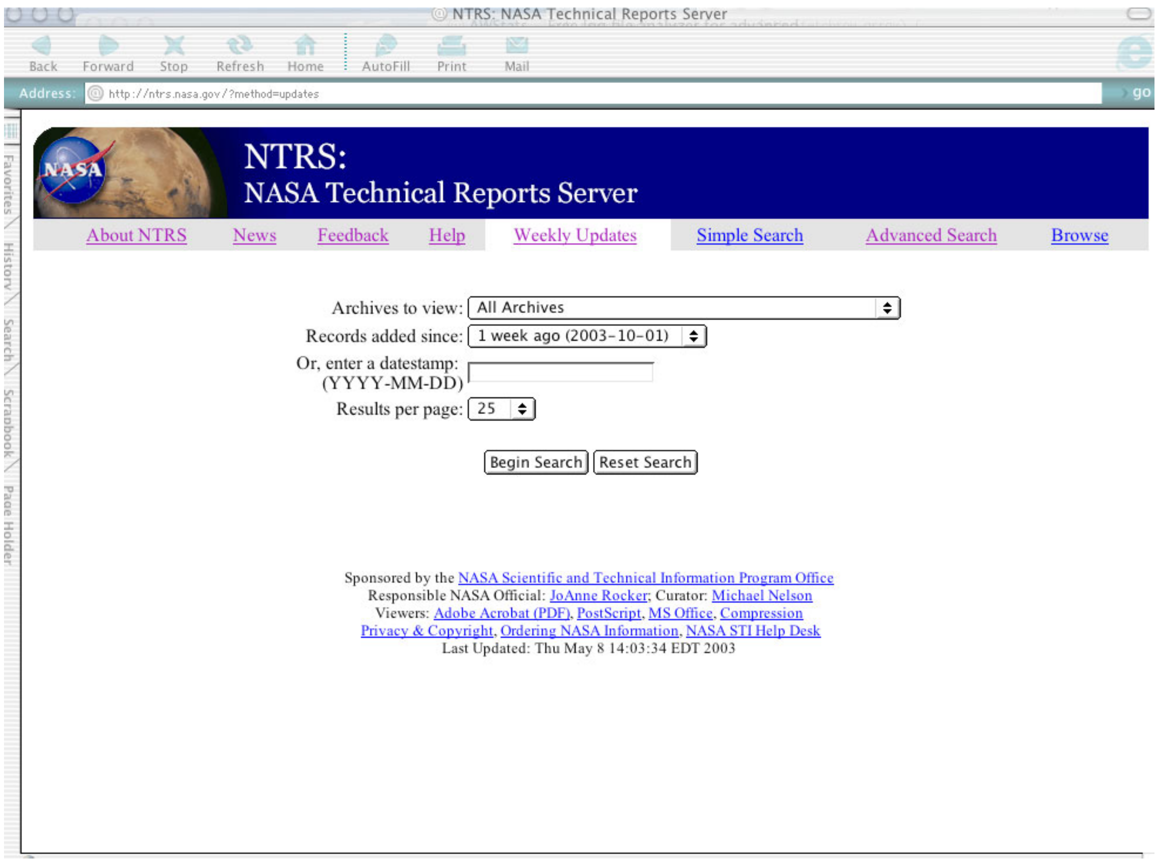
About NTRS | News | Feedback | Help | Weekly Updates | Simple Search | Advanced Search | Browse

searching for "rocker" | displaying records 1 - 25 of a total of 35 | next 25 | last 25

1. Technical Report Interchange Through Synchronized OAI Caches
Xiaoming Liu; Kurt Maly; Mohammad Zubair; Rong Tang; Mohammed Imran Padshah; George Roncaglia; JoAnne ***Rocker****; Michael Nelson; William Von Ofenheim; Richard Luce; Jacqueline Stack; Frances Knudson; Beth Goldsmith; Irma Holtkamp; Miriam Blake; Jack Carter; Mariella Di Giacomo; Major Jerome Nutter; Susan Brown; Ron Montbrand; Sally Landenberger; Kathy Pierson; Vince Duran; Beth Moser*
NASA Langley Research Center
6th European Conference on Research and Advanced Technology for Digital Libraries, Italy, Rome, September 16-18, 2002
The Technical Report Interchange project is a cooperative experimental effort between NASA Langley Research Center, Los Alamos National Laboratory, Air Force Research Laboratory, Sandia National Laboratory and Old Dominion University to allow for the integration of technical reports. This is accomplished using the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) and having each site cache the metadata from the other participating sites. Each site also implements additional software to ingest the OAI-PMH harvested metadata into their native digital library (DL). This allows the users at each site to see an increased technical report collection through the familiar DL interfaces and take advantage of whatever valued added services are provided by the native DL.
ftp://techreports.larc.nasa.gov/pub/techreports/larc/2002/mtg/NASA-2002-6ecratdl-xl.ps.Z
http://techreports.larc.nasa.gov/ltrs/PDF/2002/mtg/NASA-2002-6ecratdl-xl.pdf
Recommendations for Related Documents
Updated/Added to NTRS: 2003-04-22

2. The NASA Scientific and Technical Information (STI) Program's Implementation of Open Archives Initiative (OAI) for Data Interoperability and Data Exchange
JoAnne ***Rocker****; George J. Roncaglia; Lynn N. Heimerl; Michael L. Nelson*
NASA Langley Research Center
SLA 2002 Annual Special Libraries Association Conference , June 8-13, 2002
Interoperability and data-exchange are critical for the survival of government information management programs. E-government initiatives are transforming the way the government interacts with the public. More information is to be made available through web-enabled technologies. Programs such as the NASA s Scientific and Technical Information (STI) Program Office are tasked to find more effective ways to disseminate information to the public. The NASA STI Program is an agency-wide program charged with gathering, organizing, storing, and disseminating NASA-produced information for research and public use. The program is investigating the use of a new protocol called the Open Archives Initiative (OAI) as a means to improve data interoperability and data collection. OAI promotes the use of the OAI harvesting protocol as a simple way for data sharing among repositories. In two separate initiatives, the STI Program is implementing OAI. In collaboration with the Air Force, Department of Energy, and Old Dominion University, the NASA STI Program has funded research on implementing the OAI to exchange data between the three organizations. The second initiative is the deployment of OAI for the NASA technical report server (TRS) environment. The NASA TRS environment is

Figure A1.9. Search Results Interface.



NTRS:
NASA Technical Reports Server

Archives to view: All Archives
Records added since: 1 week ago (2003-10-01)
Or, enter a datestamp: (YYYY-MM-DD)
Results per page: 25

Sponsored by the NASA Scientific and Technical Information Program Office
Responsible NASA Official: JoAnne Rocker; Curator: Michael Nelson
Viewers: Adobe Acrobat (PDF), PostScript, MS Office, Compression
Privacy & Copyright, Ordering NASA Information, NASA STI Help Desk
Last Updated: Thu May 8 14:03:34 EDT 2003

Figure A1.10. Weekly Updates Interface.

Appendix 2: Bucket index.cgi Source Code

```
#!/usr/local/bin/perl
# Bucket "lid" - m.l.nelson@larc.nasa.gov
#
# based on various prototypes by and discussion with:
# Del Croom <d.r.croom@larc.nasa.gov>, Dan Page <d.l.page@larc.nasa.gov>
# David Bianco <djbianco@yahoo.com>, Xiaoming Liu <liu_x@cs.odu.edu>
# Hesham Farouk Anan <anan@cs.odu.edu>, Ajoy Ranga <ranga@cs.odu.edu>

# MAIN
{
        &init;
        $input = &ReadParse;

        # if no input, default method is simple -- mln (ntrs)
        if (!$input) {
                $in{"method"} = "simple";
        }

        if ($in{"verb"}) {
                $in{"method"} = $in{"verb"};
        }

        &readin_metadata unless ($in{"method"} eq "search");
        &call_method;

}

########################################################################
# Subroutine: init
# Purpose:    lots of initialization stuff...
# Created:    07/27/98  MLN
########################################################################

sub init {

        # we need perl 5.  don't do a "require 5.000" b/c no http header
        # has been printed yet (& the header is method dependent)

        if ($] < 5) {
                &complain("Perl 5 or better is needed; could only find version $]");
        }

        # pre-defined state/pref/version variables
        # note: logging variables take their name from their respective log
        $version = "_version.txt";
        $preferences{"access.log"} = "_access.log.txt";
        #$preferences{"expanding"} = "_expanding.txt";
        $preferences{"framable"} = "_framable.txt";
        #$preferences{"bcs_server"} = "_bcs_server.txt";
        #$preferences{"sfx_server"} = "_sfx_server.txt";
        $preferences{"passwd"} = "_passwd.txt";
        $preferences{"group"} = "_group.txt";
        $preferences{"host_group"} = "_host_group.txt";
        $preferences{"addr_group"} = "_addr_group.txt";
        $preferences{"method_server"} = "_method_server.txt";
        $preferences{"tc_server"} = "_tc_server.txt";
        $preferences{"maxdata"} = "_maxdata.txt";
        #$preferences{"thumbnail_increment"} = "_thumbnail_increment.txt";
        $preferences{"sendmail"} = "_sendmail.txt";

        # global file name variables

        $metadata_dir = "_md.pkg";
        $log_dir = "_log.pkg";
        $log_file = "$log_dir/access.log";
        $bucket_source_file = "index.cgi";
        $mime_file = "_http.pkg/mime.e";
        $encoding_file = "_http.pkg/encoding.e";
        $cgilib_file = "_http.pkg/cgi-lib.pl";
        $state_dir = "_state.pkg";
        $pref_dir = "_state.pkg";     # for the moment, use the state dir
```

```
# thumbnail / scan stuff
$thumbnail_increment = &internal_read_state($preferences{"thumbnail_increment"});

# lots of things could be factored out of the bucket...
$tc_dir = &internal_read_state($preferences{"tc_server"});
 if ( ($tc_dir eq "internal") || !(-d $tc_dir) ) {
        $tc_dir = "_tc.pkg";
}
$method_dir = &internal_read_state($preferences{"method_server"});
if ( ($method_dir eq "internal") || !(-d $method_dir) ) {
        $method_dir = "_methods.pkg";
}
$passwd_dir = &internal_read_state($preferences{"passwd"});
if ( ($passwd_dir eq "internal") || !(-d $passwd_dir) ) {
        $passwd_dir = $tc_dir;
}
$group_dir = &internal_read_state($preferences{"group"});
if ( ($group_dir eq "internal") || !(-d $group_dir) ) {
        $group_dir = $tc_dir;
}
$host_dir = &internal_read_state($preferences{"host_group"});
if ( ($host_dir eq "internal") || !(-d $host_dir) ) {
        $host_dir = $tc_dir;
}
$addr_dir = &internal_read_state($preferences{"addr_group"});
if ( ($addr_dir eq "internal") || !(-d $addr_dir) ) {
        $addr_dir = $tc_dir;
}
$principal_file = "$passwd_dir/passwd";
$group_file = "$group_dir/group";
$host_group_file = "$host_dir/host_group";
$addr_group_file = "$addr_dir/addr_group";

# for when we are running 1 directory down

push(@INC,"..");

# stuff for file upload...
# take from S. E. Brenner's cgi-lib.pl
# http://cgi-lib.stanford.edu/
# an uploaded element file will be stored in $in{'upfile'}
# should we do a "use CGI;" instead? -- mln

require "$cgilib_file" || &complain("could not load $cgilib_file");
$cgi_lib'maxdata = &internal_read_state($preferences{"maxdata"});

# tokens
$author_delimiter = ":::";
$myurlhere = "MYURLHERE";
$expected_access_url = "NCSTRLPLUS_URL::";

# mime & encoding

require "$mime_file";
require "$encoding_file";

# set STDERR to be STDOUT; so any errors will print to STDOUT...
open(STDERR,">&STDOUT");
select(STDOUT); $|=1;

# terms and conditions stuff
$restricted = "restricted/";
$dummy_user = "_Phil_is_Cool_";
$template_htaccess = "$tc_dir/htaccess.template";
$real_htaccess = "$tc_dir/.htaccess";
$noaccess_htaccess = "$tc_dir/htaccess.noaccess";
$entire_bucket = "index";     # "method" for protecting entire bucket

# if we are the restricted version, we need to go up 1 directory

if (&running_restricted) {
        chdir ("..");
}

use Cwd;
$real_current_dir = cwd();

# don't stop working if these libraries are not available
 eval {require LWP::Simple;}; $no_lwp = 1 if $@;
 import LWP::Simple;
```

```
        eval {require URI::URL;}; $no_lwp = 1 if $@;
        import URI::URL;
        eval {require LWP::UserAgent;}; $no_lwp = 1 if $@;
        import LWP::UserAgent;
        eval {require HTML::Entities;}; $no_html = 1 if $@;
        import HTML::Entities;
}

#########################################################################
# Subroutine:  fix_htaccess
# Purpose:     the .htaccess file for http user authentication requires
#              an absolute pathname for the passwd file.  of course,
#              absolute pathnames are the bane of bucketness.
#              so we copy a template .htaccess to the real .htaccess
#              replacing a token with the current dir
# Called:      init, tc
# Created:     08/19/1998 MLN
# Updated:     12/05/1999 MLN
# Updated:     03/23/2000 MLN
#########################################################################

sub fix_htaccess {
       my($passwd_file);      # added for support of either local or
                              # global directory for bucket passwds

       $passwd_server = &internal_read_state($preferences{"passwd"});
       if ($passwd_server =~ /internal/) {
              $passwd_file = "AuthUserFile $real_current_dir/$principal_file\n";
       } else {
              # right now, only have file system semantics
              # in the future, this should be server based - mln
              $passwd_file = "AuthUserFile $principal_file\n";
       }

       # currently, we don't use http groups, so modify only AuthUserFile

       open (H,"$template_htaccess") ||
              &complain("cannot open $template_htaccess $!");
       while (<H>)
       {
              $line = $_;
              if ($line =~ /AuthUserFile/) {
                     $line = $passwd_file;
              }
              push(@htaccess_lines,$line);
       }
       close (H);

       open (H,">$real_htaccess") ||
              &complain("cannot open $real_htaccess $!");
       chmod (0666, $real_htaccess);
       foreach $line (@htaccess_lines) {
              print H "$line";
       }
       close (H);
}

#########################################################################
# Subroutine:  running_restricted
# Purpose:     returns 1 if we are running as restricted, 0 if not
# Called:      init, tc
# Created:     08/19/1998 MLN
#########################################################################

sub running_restricted {
        my($base) = &MyBaseUrl;

        if ($base =~ /$restricted/) {
              return(1);
        }
       else {
              return(0);
       }
}

#########################################################################
# Subroutine:  readin_metadata
# Purpose:     readin the metadata file, populate the global @BIBFILE
#              the .bib file is canonical
# Called by:    main
# Created:     08/06/1997  Del Croom (d.r.croom@larc.nasa.gov)
# Updated:     10/04/1998 MLN
# Updated:     06/02/1999 MLN
########################################################################

sub readin_metadata {
        opendir(DIR,"$metadata_dir") ||
                &complain("cannot open $metadata_dir $!");
        @files = grep (/\.bib$/, (readdir(DIR)));

        if (@files) {
                $BIBFILE = $files[0];  # grab the 1st bib file
        } else {
                &complain("cannot find a RFC 1807 bib file!");
        }

        closedir(DIR);
        $BIBFILE = "$metadata_dir/$BIBFILE";

        # RFC 1807 (.bib) is the canonical format for buckets, based on
        # our early association w/ Dienst
        # read in the .bib file into @BIBFILE

        open(BIB,$BIBFILE) ||
                &complain("cannot open $BIBFILE $!");
        while(<BIB>) {
                push(@BIBFILE,$_);
        }
        close(BIB);
}

########################################################################
# Subroutine:  read_tc_file
# Purpose:     reads in the tc file, populates @TC
# Called by:   tc
# Created:     08/17/98  MLN
# Updated:     12/26/1999  MLN
########################################################################

sub read_tc_file {
        my($tc_file) = @_;

        open(T,"$tc_file") || &complain("cannot open $tc_file");
        while (<T>) {
                $line = $_;
                chomp ($line);
                push(@TC,$line);
                if ($line =~ /^user:/) {
                        $line =~ s/^user://;
                        push (@TC_USERS,$line);
                } elsif ($line =~ /^group:/) {
                         $line =~ s/^group://;
                        &read_group_file($line,"user");
                } elsif ($line =~ /^host:/) {
                         $line =~ s/^host://;
                         push (@TC_HOSTS,$line);
                } elsif ($line =~ /^host_group:/) {
                         $line =~ s/^host_group://;
                        &read_group_file($line,"host");
                } elsif ($line =~ /^addr:/) {
                         $line =~ s/^addr://;
                         push (@TC_ADDRS,$line);
                } elsif ($line =~ /^addr_group:/) {
                         $line =~ s/^addr_group://;
                        &read_group_file($line,"addr");
                } elsif ($line =~ /^inform:/) {
                         $line =~ s/^inform://;
                         push (@TC_INFORM,$line);
                } elsif ($line =~ /^package:/) {
                         $line =~ s/^package://;
                         push (@TC_PACKAGES,$line);
                } elsif ($line =~ /^element:/) {
                         $line =~ s/^element://;
                         push (@TC_ELEMENTS,$line);
                }
                # ignore lines that don't begin with:
                # user:, group:, host:, addr:, package:, element:,
                # host_group:, addr_group:
        }
        close (T);
}
```

```
########################################################################
# Subroutine:  read_group_file
# Purpose:     reads in the group file, converts the group to valid users
#              or addrs or hosts
# Called by:   read_tc_file
# Created:     12/26/1999  MLN
# Updated:     02/23/2000  MLN
# Updated:     03/20/2000  MLN
########################################################################

sub read_group_file {
        my($group,$which) = @_;
        my($file);

        if ($which =~ /addr/) {
                $file = $addr_group_file;
        } elsif ($which =~ /host/) {
                $file = $host_group_file;
        } else {
                # should not need this default
                $file = $group_file;
        }

        open(G, "$file") || &complain("cannot open $file");
        while (<G>) {
                # this does not handle a group spilling over a single
                # physical line
                $line = $_;
                chomp($line);

                # this loop reads the opened group file, looking for a match
                # to the group passed in.  when a match is found, we push all
                # values into @TC_USERS, @TC_HOSTS, @TC_ADDRS as appropriate
                if ($line =~ /^$group:/) {
                        $line =~ s/^$group://;
                        @values = split(' ', $line);
                        foreach $v (@values) {
                                if ($which =~ /user/) {
                                        push (@TC_USERS,$v);
                                } elsif ($which =~ /addr/) {
                                        push (@TC_ADDRS,$v);
                                } elsif ($which =~ /host/) {
                                        push (@TC_HOSTS,$v);
                                }
                        }
                }
        }
        close(G);
}

########################################################################
# Subroutine:  enforce_tc
# Purpose:     does the actual checks on the @TC arrays
# Called by:   tc
# Created:     08/18/98  MLN
########################################################################

sub enforce_tc {

        my($user,$host,$addr,$package,$element) = @_;

        # you can protect:
        #       1. a method
        #       2. an entire package
        #       3. a package/element combo
        #
        # note: you *cannot* combine cases 2 and 3.  either restrict entire
        #       packages, or restrict package/elements.  combinations will
        #       not work...  this is probably a bug. -- mln
        #
        # you can restrict on:
        #       1. user
        #       2. hostname
        #       3. ip address
        #
        # note: the hosts specified in the .tc file must be FQDN
        #       also, we currently don't have a canonicalization routine
        #       to handle aliases for hosts...

        # case 1: an entire method is protected
        #         there will be no values in @TC_PACKAGES or @TC_ELEMENTS
```

```
#print "host = $host";
        if (!(@TC_PACKAGES) && !(@TC_ELEMENTS) )
        {
                if (
                        grep(/$user/,@TC_USERS) &&
#                       (grep {$host =~ /$_/} @TC_HOSTS ) &&
                        (grep {$addr =~ /$_/} @TC_ADDRS )
                   )
                {
                        return (1);
                }
                else
                {
                        return (0);
                }

        } # end case 1

        # case 2: an entire package is protected.  if someone is invoking
        #         this method w/ a different package, let them pass
        #         they will be requesting a package, but no @TC_ELEMENTS
        #         will be specified

        if ( ($package) && !(@TC_ELEMENTS) )
        {
                if (grep(/$package/,@TC_PACKAGES))
                {
                        # if we're here, then $package is in @TC_PACKAGES
                        # meaning its protected
                        if (
                                grep(/$user/,@TC_USERS) &&
                                (grep {$host =~ /$_/} @TC_HOSTS ) &&
                                (grep {$addr =~ /$_/} @TC_ADDRS )
                           )
                        {
                                return (1);
                        }
                        else
                        {
                                return (0);
                        }
                }
                else
                {
                        # if we're here, then $package is not protected
                        # so let it pass through
                        return (1);
                }

        } # end case 2

        # case 3: protect an individual element and package combination
        #         if either don't match the protected list, let it pass

        if ( ($package) && ($element) )
        {
                if ( (grep(/$package/,@TC_PACKAGES)) &&
                     (grep(/$element/,@TC_ELEMENTS)) )
                {
                        # the package *and* element are on the protected lists
                        if (
                                grep(/$user/,@TC_USERS) &&
                                (grep {$host =~ /$_/} @TC_HOSTS ) &&
                                (grep {$addr =~ /$_/} @TC_ADDRS )
                           )
                        {
                                return (1);
                        }
                        else
                        {
                                return (0);
                        }
                }
                else
                {
                        # one or both of $package & $element are not protected
                        # just pass through
                        return (1);
                }
        } # end case 3

        # let's try a default value of pass...  have we missed any cases?
        return(1);
}

########################################################################
# Subroutine: tc
# Purpose:    checks terms and conditions for each of the methods
# Called by call_method
# Created:    07/18/98  MLN
########################################################################

sub tc {
        my($method) = @_;
        my($package) = $in{"pkg_name"};
        my($element) = $in{"element_name"};
        my($user) = $ENV{'REMOTE_USER'};
        my($host) = $ENV{'REMOTE_HOST'};
        my($addr) = $ENV{'REMOTE_ADDR'};

        $tc_file = "$tc_dir/$method.tc";

        if (-f $tc_file)
        {
                # see what is in the TC file
                &read_tc_file($tc_file);

                # if host or addr empty, add current values to
                # @TC_HOSTS, @TC_ADDRS so these arrays will always have
                # values (makes some of the logic easier)
                # similarly for @TC_USERS -- mln (7/1/99)

                if (!(@TC_HOSTS)) {
                        push(@TC_HOSTS,$host);
                }
                if (!(@TC_ADDRS)) {
                        push(@TC_ADDRS,$addr);
                }
                #if (!(@TC_USERS)) {
                        #push(@TC_USERS,$user);
                        #push(@TC_USERS,$dummy_user);
                #}

                # if we are a restricted version, check TC values
                if (&running_restricted)
                {
                        # this case happens when a method requires a user
                        # but the overall bucket's TC does not
                        # (it could restrict hosts only, for example) -- mln
                        if ( ($method eq $entire_bucket) &&
                             !(@TC_USERS) ) {
                                push(@TC_USERS,$dummy_user);
                                $user = $dummy_user;

                        }

                        if (&enforce_tc($user,$host,$addr,$package,$element) )
                        {
                                # everything is ok, so pass through
                        }
                        else
                        {
                                  &complain("($user, $host, $addr) not allowed for method =
$method package = $package element = $element");
                        }
                }
                else  # not running as restricted
                {
                        if (@TC_USERS)
                        {
                                # do the user restrictions match our
                                # package and/or element?
                                # note: this only does 1 p/e per method...
                                # must think of fixing that

                                if (
                                     ( !(@TC_ELEMENTS) &&
                                       grep(/$package/,@TC_PACKAGES) &&
                                       ($package)
                                     ) ||
                                  ( grep(/$package/,@TC_PACKAGES) &&
                                    grep(/$element/,@TC_ELEMENTS) &&
                                    ($package) && ($element)
                                  ) ||
                                  ( !(@TC_ELEMENTS) && !(@TC_PACKAGES) )
                                )
                              {
                                      # redirect as restricted
                                      # we must get the user info via http

                                      $restricted_url = &MyBaseUrl;
                                      $restricted_url =~ s/index.cgi//g;
                                      $restricted_url .= $restricted;
                                      if ($ENV{"QUERY_STRING"}) {
                                              $restricted_url =
"$restricted_url?$ENV{'QUERY_STRING'}";
                                      }
                                      #&fix_htaccess;
                                      print "Location: $restricted_url\n\n";

                              }
                              else  # user info not needed here
                              {
                                      # just pass through
                              }
                      }
                      else # no users defined
                      {
                              # enforce the restrictions we do have
                              # must be hosts or addrs
                              # push $dummy_user onto @TC_USERS and
                              # pass $dummy_user to &enforce_tc since
                              # any value is ok

                              if (!(@TC_USERS)) {
                                      push(@TC_USERS,$dummy_user);
                              }

                              if (&enforce_tc($dummy_user,$host,$addr,
                                  $package,$element))
                              {
                                      # everything is ok, so pass through
                              }
                              else
                              {
                                      &complain("($user, $host, $addr) not allowed for
method = $method package = $package element = $element");
                              }

                      }

              }
        }
        else  # there is no .tc file for this method
        {
                # do nothing; no TC attached to this method.
        }
}

#######################################################################
# Subroutine: call_method
# Purpose:    To invoke called method sent to script.
#             Method MUST be the first argument passed.
# Called by main
# Created:    08/06/1997  Del Croom (d.r.croom@larc.nasa.gov)
# Updated:    07/27/1998  MLN
#######################################################################

sub call_method {
        my ($method) = $in{"method"};

        # first, check for possible TC on the entire bucket
        # we'll treat this as "index.cgi" being a method
        # (implying an index.tc to specify TC)
        # then clean up @TC_USERS, @TC_HOSTS, @TC_ADDRS for a possible
        # following call to tc() for the actual method itself

        &tc($entire_bucket);
        undef(@TC_USERS); undef(@TC_ADDRS); undef(@TC_HOSTS); undef(@TC);

        $method_file = "$method_dir/$method.pl";
        if (-f $method_file) {

                # check tc; load the file; call the method
                &tc($method);

                # if we made it out of &tc, we must be ok...
                require "$method_file";
                &$method;
        }
        else {
#                &unsupported($method);
                 require "_methods.pkg/Identify.pl";
                 &Identify;
        }
}

########################################################################
# Subroutine: log
# Purpose:    Opens the bucket logfile (access.log) and appends an entry
#             as described by the calling routine
# Called by most methods
# Created 08/08/97  Del Croom (d.r.croom@larc.nasa.gov)
# Updated:     08/20/1998 MLN
# Updated:     06/04/1999 MLN
# Updated:     01/17/2000 MLN
# Updated:     04/10/2000 MLN
# Updated:     12/19/2002 MLN  converted to CLF
########################################################################

sub log {
        my($action, $status, $msg) = @_;

        if (!(defined($once))) {
                $once = 1;
        } else {
                # we're in a &log ... &complain ... &log loop
                exit 0;
        }

        # if logging is off, and there is no one to inform, just return
        if ( (&internal_read_state($preferences{"access.log"}) eq "off") &&
             !(@TC_INFORM) )
        {
                return;
        }

        # if no hostname, and no addr - it must have come from the
        # command line

        if ($ENV{'REMOTE_HOST'}) {
                $host=$ENV{'REMOTE_HOST'};
         } elsif ($ENV{'REMOTE_ADDR'}) {
                $host=$ENV{'REMOTE_ADDR'};
        } else {
                $host="(command line)";
        }

        $agent=$ENV{'HTTP_USER_AGENT'};

        @now=localtime(time);                # get current timestamp
        ($sec,$min,$hr,$day,$mon,$yr)=@now[0..5]; # extract fields from @now
        @months=('Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec');
        $yr += 1900;

        #$log_entry = sprintf "%-30s [$months[$mon]-%02d-%04d]
%02d:%02d",$host,$day,$yr,$hr,$min;
        #$log_entry .= " $action $status \"$msg\" $agent\n";

        $date_log_entry = sprintf
"[%02d/$months[$mon]/%04d:%02d:%02d:%02d]",$day,$yr,$hr,$min,$sec;
        if ($ENV{'QUERY_STRING'}) {
                $url_log_entry = "$ENV{'SCRIPT_NAME'}?$ENV{'QUERY_STRING'}";
        } else {
                $url_log_entry = "$ENV{'SCRIPT_NAME'}";
        }
        $log_entry = "$host - - $date_log_entry \"$ENV{'REQUEST_METHOD'} $url_log_entry
$ENV{'SERVER_PROTOCOL'}\" \"$ENV{'HTTP_USER_AGENT'}\"\n";


        if (&internal_read_state($preferences{"access.log"}) eq "on") {
                open(LOG,">>$log_file");
                chmod (0666, $log_file);
                print LOG "$log_entry";
                close(LOG);
        }

        # check to see if anyone needs to be informed about what just
        # happened with this bucket

        if (@TC_INFORM) {
                $fullurl = &MyFullUrl;
                $baseurl = &MyBaseUrl;
                $inform = join(',',@TC_INFORM);
                $sendmail = &internal_read_state($preferences{"sendmail"});
                # quietly quit if we can't send mail
                #open(MAIL, "|$sendmail") || return (0);
                open(MAIL, "|$sendmail") || die "no senmail\n";
                print MAIL "From: $inform\n";
                print MAIL "To: $inform\n";
                print MAIL "Subject: $in{\"method\"} on $baseurl\n";
                print MAIL "\n\n$log_entry\n\n";
                print MAIL "method = $in{\"method\"} \n\n";
                print MAIL "full URL = $fullurl \n\n";
                print MAIL "error message = $msg \n\n";
                if ($status eq "OK") {
                        print MAIL "succeeded -- no action required.\n";
                } elsif ($no_html) {
                        print MAIL "did not succeed. This bucket cannot ensure proper
encoding of the attempted URL to allow you to retry.\n";
                } else {
                        print MAIL "did not succeed.  An HTML form is included below to
allow a retry of this action.\n\n\n";
                        print MAIL "<form method='POST' enctype='multipart/form-data'
action='$baseurl'>\n";
                        print MAIL "<table border=0>\n";
                        foreach $k (keys %in) {
                                $value = HTML::Entities::encode($in{"$k"});
                                print MAIL "<tr><td>$k <td><input type=textbox name=$k
value=\"$value\">\n";
                        }
                        print MAIL "</table>\n";
                        print MAIL "<input type=submit value=\"Approve this
action\"></form>\n";
                }
                print MAIL "\n.\n";
                close(MAIL);
        }

}

########################################################################
# Subroutine:           http_header
# Use:          sets the MIME type
# Updated: 07/27/1998 MLN
########################################################################

sub http_header {
        my ($type, $encoding) = @_;

        print "Content-type: $type\n";
        if ($encoding) {
                print "Content-encoding: $encoding\n";
        }
        print "\n";

}

########################################################################
# Subroutine:           file_size
# Use:          return the filesize
# Called by:    display_default
# Created:      08/04/1998 MLN
# Updated:      06/05/1999 MLN
########################################################################

sub file_size {
        my($file) = @_;
        my($bytes);
        my($kb) = 1024;
        my($mb) = 1000 * $kb;
        my($gb) = 1000 * $mb;
        # if readable file, and a plain file...
        if ( (-r $file) && (-f $file) ) {
                $bytes = (-s $file);
                # pretty print file sizes
                if ($bytes > $gb) {
                        $bytes = int($bytes / $gb);
                        return "($bytes GB)";
                }
                elsif ($bytes > $mb) {
                        $bytes = int($bytes / $mb);
                        return "($bytes MB)";
                }
                elsif ($bytes > $kb) {
                        $bytes = int($bytes / $kb);
                        return "($bytes KB)";
                }
                else {
                        return "($bytes bytes)";
                }
        } else {
                # probably a redirect -- no file size
                return ("");
        }
}


########################################################################
# Subroutine:  shorten_bibfile
# Use:     deletes package or element info from the bib file
# Created: 07/27/1998 MLN
########################################################################

#sub shorten_bibfile {
#       my($bibfile,$type,$name) = @_;

#       open (BIB, ">$bibfile") ||
#               &complain("cannot open $bibfile $!");
#       chmod (0666,$bibfile);

#       foreach $line (@BIBFILE) {
#               if (($line =~ /^$type-END::/) && ($line =~ /$name/) )
#               {
#                       $skip = 0;
#                       next;
#               }
#               if (($line =~ /^$type/) && ($line =~ /$name/) )
#               {
#                       $skip = 1;
#                       next;
#               }
#               if ($skip) {
#                       next;
#               } else {
#                       print BIB "$line";
#               }
#       }
#       close (BIB);
#}

########################################################################
# Subroutine:           append_bibfile
# Use:          Adds new info to the bibfile
# Created:      07/26/98 MLN
# Updated:      07/06/1999 MLN
########################################################################

#sub append_bibfile {
#         my ($bibfile, $newinfo,$endtag) = @_;

#       lock($BIBFILE);
#       open (BIB, ">$bibfile") ||
#               &complain("cannot open $bibfile $!");
#       chmod (0666,$bibfile);
#         $newinfo =~ s/\r//g;          # from Ajoy, 9/18/98

#       foreach $line (@BIBFILE)
#       {
#               $line =~ s/\r//;        # from Ajoy, 9/18/98
#               if ($line =~ /$endtag/) {
#                       # this should handle both new packages, and
#                       # elements being placed within a package
#                       print BIB "$newinfo\n";
#                       print BIB "$line";
#               } else {
#                       print BIB "$line";
#               }
#       }

#       unlock($BIBFILE);
#}

#########################################################################
# Subroutine:           lock
# Use:          creates a separate lock file
# Updated:      07/06/1999 MLN
#########################################################################

#sub lock {
#       my ($file) = @_;

#       $LOCK_EX = 2;  # exclusive lock

#       open(FH,"$file") || &complain("cannot open $file $!");
#       flock (FH, $LOCK_EX);
#       close(FH);
#}

#########################################################################
# Subroutine:           unlock
# Use:          unlocks a lock file
# Updated:      07/06/1999 MLN
#########################################################################

#sub unlock {
#       my ($file) = @_;
#
#       $LOCK_UN = 8;  # unlock
#
#       open(FH,$file) || &complain("cannot open $file $!");
#       flock ($FH, $LOCK_UN);
#       close(FH);
#}

#########################################################################
# Subroutine:           unsupported
# Use:          Notifies requester that an unsupported method was called
# Updated:      06/05/1999 MLN
#########################################################################

sub unsupported {
        my($method) = @_;
        &http_header("text/plain");

        print "bucket method \"$method\" not found or implemented\n";

        &log("unsupported","ERR","$in{'method'} not implemented");
}

#########################################################################
# Subroutine:   name_collision
# Use:          returns unique names
# Created:      9/13/98 MLN
#########################################################################

#sub name_collision
#{
#       my ($pkg_name, $element_name) = @_;
#       my ($counter) = 0;
#
#       while (-e "$pkg_name/$element_name") {
#               $element_name =~ s/\.pkg$//g;
#               $element_name =~ s/\.$counter$//g;
#               $counter += 1;
#               $element_name .= ".$counter.pkg";
#       }
#
#       return ($element_name);
#}

#########################################################################
# Subroutine:  complain
# Use:         a nasty gram to the user
# Created:     7/28/98 MLN
########################################################################

sub complain
{
       my($msg) = @_;

       if (!($already_printed_header)) {
              $already_printed_header = 1;
              &http_header("text/plain");
       }

       print ("$msg\n");
       &log("complain","ERR","$msg");
       exit (0);
}

########################################################################
# Subroutine:   internal_read_state
# Purpose:      returns the value of the state passed in.  NOTE: this is
#              different than displaying the element for that state --
#              that value gets passed back to the client
# Created:      06/30/1999  MLN
# Updated:     03/20/2000 MLN
########################################################################

sub internal_read_state {
       my($state) = @_;
       my($value);

# hardwired for speed in NTRS

       if ($state eq "_access.log.txt") {
              return ("on");
       } elsif ($state eq "_framable.txt") {
              return ("off");
       } elsif ($state eq "_passwd.txt") {
              return ("internal");
       } elsif ($state eq "_group.txt") {
              return ("internal");
       } elsif ($state eq "_host_group.txt") {
              return ("internal");
       } elsif ($state eq "_addr_group.txt") {
              return ("internal");
       } elsif ($state eq "_method_server.txt") {
              return ("internal");
       } elsif ($state eq "_tc_server.txt") {
              return ("internal");
       } elsif ($state eq "_sendmail.txt") {
              return ("/usr/lib/sendmail -t");
       } elsif ($state eq "_maxdata.txt") {
              return ("5000000");
       }

       # the state dir is either below us, or up one and down if we're
       # running restricted
#      if (-f "../$state_dir/$state") {
#             open(S,"../$state_dir/$state") ||
#                    &complain("cannot open ../$state_dir/$state $!");
#      } elsif (-f "$state_dir/$state") {
#             open(S,"$state_dir/$state") ||
#                    &complain("cannot open $state_dir/$state $!");
#      } elsif (-f "$real_current_dir/$state_dir/$state") {
#             open(S,"$real_current_dir/$state_dir/$state") ||
#                    &complain("cannot open $state_dir/$state $!");
#      } else {
#             &complain("internal_read_state cannot read $state $!");
#      }
#
#      while (<S>) {
#             # most options will be a single line
#             $value .= $_;
#      }
#      chomp $value;
#      return ($value);
}

########################################################################
# Subroutine:   isaurl
# Purpose:      returns true if the argument appears to be an absolute
#               URL, false if otherwise
# Called by:    display
# Created:      08/03/98  MLN
#########################################################################

sub isaurl
{
        my($check) = @_;
        @urls = ("http://", "ftp://", "gopher://", "mailto://", "wais://", "http
s://");

        foreach $u (@urls) {
                if ($check =~ /$u/) { return(1) };
        }
        return (0);
}

#########################################################################
# Subroutine:  send_mesg
# Use:         sends the message stored in "%mesg"
# Created:     03/24/2000 MLN
#########################################################################

sub send_mesg {
        my($mesg_target) = @_;
        if ($no_lwp) {
                return ("this bucket cannot send messages");
        }

        $ua = new LWP::UserAgent;
        my $req = new HTTP::Request 'POST', "$mesg_target";
        # use this as a hook for future auth capability
        # $req->authorization_basic('name', 'mypassword');
        $req->content_type('application/x-www-form-urlencoded');
        my $curl = url("http:");      # create an empty HTTP URL object
        $curl->query_form(%mesg);
        $req->content($curl->equery); # %mesg content as escaped query string
        $result = $ua->request($req);

        if ($result->is_success) {
                return($result->content);
        } else {
                return("");
        }
}

#########################################################################
# Subroutine:   db_error
# Use:          come here if the database is down
# Created:      04/15/03 MLN
#########################################################################

sub db_error {
        my($err_msg) = @_;

        print <<EOF;

<p>
<blockquote>
<table border=0>
<tr>
<td bgcolor=dedede>
We're sorry, the database for the NASA Technical Report Server is
currently unavailable.  The NTRS administrators have been notified,
and service should be restored shortly.
</table>
</blockquote>
<p>

EOF

print $footer;
push(@TC_INFORM,"ntrs+admin\@larc.nasa.gov");
&log("about","OK","$err_msg");
exit;

}

# END
```

Appendix 3: NTRS OAI Source Code

```
::::::::::::::
_methods.pkg/ListIdentifiers.pl
::::::::::::::
#########################################################################
# Method:       ListIdentifiers
# Use:                 OAI; returns the "identifiers" in a list
#              for now, identifier = report #
# Created:      11/03/2000 MLN
# Updated:     01/11/2001 MLN
# Updated:     10/04/2001 MLN - added set support
# Updated:     02/15/2002 MLN - added resumptionToken support
# Updated:      06/17/2002 TLH - made OAI2.0 compliant
# Note:                        (setSpec inserted for NTRS implementation)
# Updated:     07/08/2002 TLH - bug fixes :o)
# Updated:     07/18/2002 TLH - @records = &get_ids; now called in oai.pl
#########################################################################

sub ListIdentifiers
{
        my ($from) = $in{"from"};
        my ($until) = $in{"until"};
        my ($set) = $in{"set"};
        my ($resumption) = $in{"resumptionToken"};
        my ($datestamp,$id,$start_range,$end_range,$current_record,
               $total_records);

        require "oai/oai.pl";
        &too_busy;     # redirect to another server?

         $current_record = 0;
         $total_records = $#records + 1;

         if ($resumption) {
                 # decode old resumption token
                 ($from,$until,$metadataPrefix,$set,$start_range,$end_range) =
                         &decode_resumption($resumption);

                 # get new resumption token
                 $next_resumption = &get_next_resumption($resumption);

         } else {
                 # first time called
                 $start_range = 0;
                 $end_range = $max_records_to_return;
         }

        &http_header("text/xml");

        # print XML response
        print "$first_xml_line\n";
        print "$pmh_xml_line\n";
        print "$time_xml_line\n";
        print "$request_xml_line\n";

         if (length($errors) )
             {print $errors; }



        # ---------------
        # Here, $result will gather up the response, to be printed if \
        #        $current_record is > 0
         # ---------------

        else
        {   $result = "";
            $result .= "$verb1_xml_line\n";

            foreach $id (@records) {
               $datestamp = &get_datestamp($id);

               if ($from) {
                      next unless ($from le $datestamp);
               }
               if ($until) {
```

```
                        next unless ($until ge $datestamp);
                }

                 if ($set) {
                        # assume set value is correct
                        # happily return no results on bad set values
                        next if (!&set_in_metadata($set,$id));
                }

                $current_record++;
                 next if ($current_record < $start_range);

                $result .= "<header>\n";
                $result .=  "<identifier>$id</identifier>\n";
                $result .= "<datestamp>$datestamp</datestamp>\n";

                $result .= &set_membership_of_record($id);
                $result .= "</header>\n";

                 if ($current_record == $total_records) {
                        # done - quit and flag resumption
                        $next_resumption="makeMeEmpty";
                        last;
                 }
                 if ($current_record == $end_range) {
                        # time to quit this response
                        if (!($next_resumption)) {
                                # make a resumptionToken if we don't have
                                # one from above
                                $next_resumption = &encode_resumption($from,
                                $until,$metadataPrefix,$set,
                                $start_range + $max_records_to_return + 1,
                                $max_records_to_return * 2 );
                        }
                        last;
                 }

        }

        # ----------------
        # NOW CHECK TO SEE IF THERE IS A NEW RESUMPTION TOKEN VALUE.  IF WE ARE AT THE
        # END OF THE RECORDS, RETURN AN EMPTY RESUMPTION TOKEN ELEMENT
        # ----------------
        if ($next_resumption eq "makeMeEmpty") {
                $result .=  "<resumptionToken/>\n";
        }
        elsif ( ($next_resumption) &&
             ($current_record == $end_range)) {
                #NOTE: completeListSize not currently implemented
#               $result .=  "<resumptionToken completeListSize=\"num of records that
match\"> \
                $result .=  "<resumptionToken>$next_resumption</resumptionToken>\n";
        }

        $result .= "$verb2_xml_line\n";
        } #end of else


        # ----------------
        # NOW CHECK TO SEE IF THERE ARE ANY VALID RECORDS; IF SO PRINT $result;
        # IF NOT, PRINT NO_RECORDS ERROR
        # ----------------
        if ($current_record > 0)
          { print "$result"; }
# Note: other implementations use $current_record > 1
#{print "<valid_records>$current_record</valid_records> \n"; }
        else
          { print "<error code=\"noRecordsMatch\">No Records Match</error>\n";}


        print "$pmh2_xml_line\n";
        &log("ListIdentifiers","OK","$resumption");
}

1;
::::::::::::::
_methods.pkg/ListMetadataFormats.pl
::::::::::::::
########################################################################
# Method:       ListMetadataFormats
# Use:                  OAI; returns the "sets" define by the DL/archive
# Note:        assumes uniform availability of metadata formats
#              for all records (may not always be valid)
# Created:      11/03/2000 MLN
# Updated:     01/14/2001 MLN
# Updated:      06/19/2002 TLH       Converted to OAI 2.0 Protocol
#########################################################################

sub ListMetadataFormats
{
        my ($id) = $in{"identifier"};
        require "oai/oai.pl";

        &http_header("text/xml");

        # print XML response
        print "$first_xml_line\n";
        print "$pmh_xml_line\n";
        print "$time_xml_line\n";
        print "$request_xml_line\n";

        if (length($errors) )
            {print $errors; }

        else {
               print "$verb1_xml_line\n";
               @formats = &get_metadata_formats($id);
               foreach (@formats) {
                       print "$_\n";
               print "$verb2_xml_line\n";
               }
        }

        print "$pmh2_xml_line\n";

        &log("ListMetadataFormats","OK","-");
}

1;
::::::::::::::
_methods.pkg/ListRecords.pl
::::::::::::::
#########################################################################
# Method:       ListRecords
# Use:                 OAI; returns the metadata records
# Created:      11/03/2000 MLN
# Updated:      10/04/2001 MLN - added set support
# Updated:     02/15/2002 MLN - added resumptionToken support
# Updated:     06/19/2002 TLH - converted to OAI2.0
# Updated:     07/11/2002 TLH - converting for NTRS use
# Modified:    07/17/2002 TLH - modified set_membership_of_records
# Modified:    07/18/2002 TLH - @records=&get_ids now called in oai.pl
#########################################################################

sub ListRecords
{
        my ($from) = $in{"from"};
        my ($until) = $in{"until"};
        my ($resumption) = $in{"resumptionToken"};
        my ($set) = $in{"set"};
        my ($datestamp,$metadata,$id,$start_range,$end_range,$current_record,
               $total_records);


        require "oai/oai.pl";

        &too_busy;      # redirect to another server?

###############
#Note  @records NOW GOTTEN IN OAI.PL
#      @records = &get_ids;
##############
        $current_record = 0;
        $total_records = $#records + 1;

        if ($resumption) {
                # decode old resumption token
                ($from,$until,$metadataPrefix,$set,$start_range,$end_range) =
                        &decode_resumption($resumption);

                # get new resumption token
                $next_resumption = &get_next_resumption($resumption);
```

```
} else {
        # first time called
        $start_range = 0;
        $end_range = $max_records_to_return;
}

&http_header("text/xml");

# print XML response
print "$first_xml_line\n";
print "$pmh_xml_line\n";
print "$time_xml_line\n";
print "$request_xml_line\n";

if (length($errors) )
    {print $errors; }


# ---------------
# Here, $result will gather up the response, to be printed if \
#        $current_record is > 0
# ---------------

else
{  $result = "";
   $result .= "$verb1_xml_line\n";


   foreach $id (@records) {
        undef($metadata);       # get rid of any old values

       $datestamp = &get_datestamp($id);

       if ($from) {
               next unless ($from le $datestamp);
        }
        if ($until) {
              next unless ($until ge $datestamp);
        }


       if ($set) {
              # assumet set value is correct
              # happily return no results on bad set values
              next if (!&set_in_metadata($set,$id));
       }

       $current_record++;
        next if ($current_record < $start_range);

        $metadata = &get_dc($id);

       $result .= "<record>\n";
       $result .= "<header>\n";
       $result .= "<identifier>$id</identifier>\n";
       $result .= "<datestamp>$datestamp</datestamp>\n";
       #ADD setSpec DATA IF IT EXISTS (done in next line)
       $result .= &set_membership_of_record($id);
       $result .= "</header>\n";
       $result .= "$metadata\n";
       $result .= "</record>\n";

        if ($current_record == $total_records) {
                # done - quit and flag resumption
                $next_resumption="makeMeEmpty";
                last;
        }

        if ($current_record == $end_range) {
                # time to quit this response

                if (!($next_resumption)) {
                        # make a resumptionToken if we don't have
                        # one from above
                        $next_resumption = &encode_resumption($from,
                        $until,$metadataPrefix,$set,
                        $start_range + $max_records_to_return + 1,
                        $max_records_to_return * 2 );
                }
                last;
                }

           } #end of foreach


           # ----------------
           # NOW CHECK TO SEE IF THERE IS A NEW RESUMPTION TOKEN VALUE.  IF WE ARE AT THE
           # END OF THE RECORDS, RETURN AN EMPTY RESUMPTION TOKEN ELEMENT
           # ----------------
           if ($next_resumption eq "makeMeEmpty") {
                   $result .=  "<resumptionToken/>\n";
           }
           elsif ( ($next_resumption) &&
                ($current_record == $end_range)) {
                   #NOTE: completeListSize not currently implemented
#                  $result .=  "<resumptionToken completeListSize=\"num of records that
match\"> \
                   $result .=  "<resumptionToken>$next_resumption</resumptionToken>\n";
           }

           $result .= "$verb2_xml_line\n";

       } #end of else


        # ----------------
        # NOW CHECK TO SEE IF THERE ARE ANY VALID RECORDS; IF SO PRINT $result;
        # IF NOT, PRINT NO_RECORDS ERROR
        # ----------------
        if ($current_record > 0)
          { print "$result"; }
        else
          { print "<error code=\"noRecordsMatch\">No Records Match</error>\n";}


        print "$pmh2_xml_line\n";
       &log("ListRecords","OK","$resumption");
}

1;
::::::::::::::
_methods.pkg/ListSets.pl
::::::::::::::
#######################################################################
# Method:       ListSets
# Use:          OAI; returns the "sets" define by the DL/archive
# Created:      11/03/2000 MLN
# Updated:      01/11/2001 MLN
# Updated:      06/19/2002 TLH - Converted to OAI2.0
########################################################################

sub ListSets
{
    require "oai/oai.pl";

    &too_busy;      # redirect to another server?

    @sets = &get_sets;

    &http_header("text/xml");

    # print XML response
    print "$first_xml_line\n";
    print "$pmh_xml_line\n";
    print "$time_xml_line\n";
    print "$request_xml_line\n";


    if (length($errors) )
        {print $errors; }
    else
    {
        print  "$verb1_xml_line\n";
        foreach (@sets) {
                next if $_ eq "";
                print "<set>\n";
                print "$_\n";
                print "</set>\n";
        }

        print "$verb2_xml_line\n";
    }
        print "$pmh2_xml_line\n";
        &log("ListSets","OK","-");
}

1;
::::::::::::::
_methods.pkg/GetRecord.pl
::::::::::::::
#########################################################################
# Method:       GetRecord
# Use:                  OAI; returns the metadata records
# Created:      11/03/2000 MLN
# Updated:     01/14/2001 MLN
# Updated:     06/19/2002 TLH - Converted to OAI 2.0 Protocol
# Modified:    07/17/2002 TLH - Modified set_membership_of_record
#                              line
#########################################################################

sub GetRecord
{
       my($id) = $in{"identifier"};
       my($resumption) = $in{"resumptionToken"};
       my($metadataPrefix) = $in{"metadataPrefix"};
       my($datestamp,$metadata);


       require "oai/oai.pl";


       &http_header("text/xml");

       # print XML response
       print "$first_xml_line\n";
       print "$pmh_xml_line\n";
       print "$time_xml_line\n";
       print "$request_xml_line\n";

        if (length($errors) )
            {print $errors; }

       # if no errors....
       else
{
               $metadata = &get_dc($id);
               $datestamp = &get_datestamp($id);
               $setStuff = &set_membership_of_record($metadata, $id);

               print "$verb1_xml_line\n";
               print "<record>\n";
               print "<header>\n";
               print "<identifier>$id</identifier>\n";
               print "<datestamp>$datestamp</datestamp>\n";
                #ADD setSpec DATA IF IT EXISTS (done in next line)
                print "$setStuff";
               print "</header>\n";
               print "$metadata\n";
               print "</record>\n";
               print "$verb2_xml_line\n";
       }

       print "$pmh2_xml_line\n";

       &log("GetRecord","OK","$id");
}

1;
::::::::::::::
_methods.pkg/Identify.pl
::::::::::::::
#########################################################################
# Method:       Identify
# Use:          OAI; returns the archive description
# Created:      11/03/2000 MLN
# Modified:     06/12/2002 TLH  (mod. to conform to OAI2.0 Spec)
#########################################################################

sub Identify
{
        require "oai/oai.pl";
        my ($response);
        &http_header("text/xml");

        # print XML response
        print "$first_xml_line\n";
        print "$pmh_xml_line\n";
        print "$time_xml_line\n";

        print $request_xml_line;

       if (length($errors) )
        {print $errors; }
       else
        {
         print "$verb1_xml_line\n";
         $response = &identify;
         print "$response\n";
         print "$verb2_xml_line\n";
        }

        print "$pmh2_xml_line\n";

        &sys_load;
        &log("Identify","OK","-");
}

1;

::::::::::::::
oai/oai.pl
::::::::::::::
########################################################################
# File: oai.pl
# Use:  library file for NTRS OAI archive
#       this file is specific to the NTRS DL
# Created:     11/03/2000 MLN
# Updated:     01/11/2001 MLN
# Updated:     07/08/2001 MLN added OAI 1.1 support
# Updated:      06/27/2002 TLH - Modified to be OAI 2.0 compliant
# Updated:     07/09/2002 TLH - Modified to work for NTRS Archive (started)
# Updated:     07/15/2002 TLH - Bug fixes var scoping in $res var error test scoping
# Updated:      07/23/2002 TLH - $max_records_to_return = 50 for calls w.
#              set arg present to avoid timeouts
# Updated:     05/13/2003 MLN - changed to use MySQL instead of the filesystem
#
# NOTES:         - repositories within may or may not support Sets.
#               - (check noSetHierarchy for functionality on diff repositories)
#                -  ListMetadatFormats : noMetadataFormats error not implemented
#                   as the dc format has been hardcoded as the response, through
#                   &get_metadata_formats   - TLH
#                -  (Earliest datestamp is a bogus hardcode for the moment (1900-01-01)
# NOTE:         - Belvedere implementation has additional code in get_ids to
#                     handle OAI1.1 dc reocords and dir naming .

######################################################################

use Text::Wrap;
use File::Find;
use File::Basename;
use Shell qw(uptime);
use DBI;

require "oai/sets.pl";

$id_db         = "oai/id.db";
$report_root    =  "/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/";
#$mybaseurl    = &encode_url(&MyBaseUrl);
$mybaseurl     = &MyBaseUrl;
$responsedate  = &get_response_date;
$oai_version   = "2.0";

$datasource = "DBI:mysql:database=ntrs;host=localhost";
$user = "mln";
$password = "ntrsr0cks";


if (    $in{'method'} eq "ListRecords" ||
        $in{'method'} eq "ListIdentifiers"  ||
        ( $in{'method'} eq "ListMetadataFormats" && $in{'identifier'} )
   )    {
               @records = &get_ids;
        }


# IDENTIFY VARS....
# This date is hardcoded in (**1900 is a temporary date**) - TLH
$dl                 = "ntrs.nasa.gov";
$earliest_datestamp = "1900-01-01T12:00:00Z";
$deleted_record      = "no";
$granularity         = "YYYY-MM-DD";
$repositoryId       = "ntrs.nasa.gov";

$first_xml_line = "<?xml version=\"1.0\" encoding=\"UTF-8\"?>";
$pmh_xml_line = "<OAI-PMH xmlns=\"http://www.openarchives.org/OAI/2.0/\"
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:schemaLocation=\"http://www.openarchives.org/OAI/2.0/
http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd\">";
$pmh2_xml_line = "</OAI-PMH>";

$request_xml_line  = &get_request_open;
$request_xml_line .= "$mybaseurl</request>\n";

$errors           = &get_error;

$verb1_xml_line    = "<$in{'method'}>";
$verb2_xml_line    = "</$in{'method'}>";
$time_xml_line     = "<responseDate>$responsedate</responseDate>\n";

$max_records_to_return = 300;


#########################################################################
# Function:    get_id_data
# Use:         To extract varible info from the id
# Created:     07/10/2002 TLH
# Note:        This function populates: $archive
#########################################################################

sub get_id_data {
        # Get $archive var w. following line...
        #($oai,$archive,$year,$report) = split(":",$id);
        &split_id($id);
}


#########################################################################
# Function:     get_request_open
# Use:          returns the <request> element open tag w. verb and args contained
#               as attributes within the tag
# Note:         end tag and element content still needed (this only creates the tag open)
# Created:      06/12/2002 TLH
#########################################################################
sub get_request_open {
        $response = "<request ";
        foreach $_ (keys(%in)) {
          $val       = $in{"$_"};
          if ($_  eq "method")
                {$verb=  $val; }
          else
                {
                $response .= " $_=\"$val\"";
                }
         }
        $response .= ">";

        return ("$response");
}



#########################################################################
# Function:     get_error
# Use:          returns the <error> element if errors are found
# Tests:        1. Is arg recognized ..
#              1.1 Is from/until formatted correctly YYYY-MM-DD ?
#               2. Is verb recognized ..
#               3. Is arg use appropriate w. verb
#               4. Presence of ResumptionToken tested for exclusivity
#               5. Verb specific tests.
# NOTE:         Special case for if($arg eq "method")  (it's ignored)
# Created:      06/17/2002 TLH
#########################################################################
```

```
sub get_error {
        $response ="";

        @args = keys(%in);


        # --------------------
        # 1. ILLEGAL_ARGUMENT (GENERIC TEST)
        #                  (THIS ALSO SETS VAR VALUES)
        # --------------------

        $count = @args +0;

        foreach $arg (@args) {
          $val = $in{"$arg"};
          if     ($arg eq "method")         {$count--;}
          elsif  ($arg eq "verb")           {$verb = "$val";}
          elsif  ($arg eq "from")           {$from = "$val";}
          elsif  ($arg eq "until")          {$until = "$val";}
          elsif  ($arg eq "metadataPrefix") {$metadataPrefix = "$val";}
          elsif  ($arg eq "resumptionToken"){$resumptionToken = "$val";}
          elsif  ($arg eq "identifier")     {$identifier = "$val";}
          elsif  ($arg eq "set")            {$set = "$val";}
          else   { $response .="<error code=\"badArgument\">Illegal Argument:
$arg</error>\n";}
        }


       # ---------------------
       # 1.1 FROM/UNTIL CASE - if they exist, are they formatted correctly?
       # ---------------------
       if ($from)
         { if (! &check_date_format("from"))
              { $response .= "<error code=badArgument>from date $from not valid
</error>\n"; }
       }

       if ($until)
         { if(! &check_date_format("until"))
               { $response .= "<error code=badArgument>until date $until not valid
</error>\n"; }
       }



        # ---------------------
        # 2. BAD_VERB CASE -  IF VERB IS NOT ONE OF THE 6 PERMITTED:
        # ---------------------
        if   (!$verb)
               { $response .="<error code=\"badVerb\">Verb required</error>\n"; }

        elsif(!( ($verb eq "Identify")  ||
                 ($verb eq "ListIdentifiers")   ||
                 ($verb eq "ListSets")  ||
                 ($verb eq "ListRecords")       ||
                 ($verb eq "ListMetadataFormats")||
                 ($verb eq "GetRecord")
              ) )
                { $response .="<error code=\"badVerb\">The request verb ($verb)\
                                 is illegal</error>\n"; }



        # ---------------------
        # RESUMPTION_TOKEN CASE
        # 1. RESUMPTION_TOKEN IS AN EXCLUSIVE ARGUMENT IN A REQUEST
        # 2. IS IT A VALID TOKEN ? (This needs further implemenation of result
processing)
        # ---------------------

        if ( ($resumptionToken) && ($count >2) )
            { $response .="<error code=\"badArgument\">The request contains an illegal
argument(s) \
                        Note: resumptionToken is an exclusive argument</error>\n"; }

        #####################################################
##NOTE: ### This should be implemented further ###
        ### This stage splits out all the args of the resumption token
        ### They are ready for processing. (Processing still needs to be done)
        if ($resumptionToken)
            {
#             my($from,$until,$metadataPrefix,$set,$start_range,$end_range);
              ($from,$until,$metadataPrefix,$set,$start_range,$end_range) =
&decode_resumption($resumptionToken);
                ### if testing yields something bad then output should be as follows...


              if ($from)
              { if (! &check_date_format("from"))
                       { $response .="<error code=\"badResumptionToken\">Illegal
Resumption \
                              Token = $resumptionToken  (test 1)</error>\n";}
              }


              if ($until)
              { if(! &check_date_format("until"))
                       { $response .="<error code=\"badResumptionToken\">Illegal
Resumption \
                              Token = $resumptionToken  (test 2)</error>\n";}
               }


              # pattern match here for presence of five "!" marks in resumptionToken
issued
              # ----------------------------------------------
              $res_match = $resumptionToken;
              $res_match =~ s/\!/ /g ;
              if (!($res_match=~ /\s.*\s.*\s.*\s.*\s.*/ ))
               {
                $response .="<error code=\"badResumptionToken\">Illegal Resumption \
                              Token = $resumptionToken (test 3)</error>\n";
               }


           }
        ######################################################




        # ---------------------
        # CANNOT_DISSEMINATE_FORMAT CASE (FOR ListIdentifiers/ListRecords/GetRecord)
        # ---------------------
       if ( $verb eq "ListIdentifiers" || $verb eq "ListRecords" || $verb eq "GetRecord"
)
         {
           if ( ($metadataPrefix)  &&  !(&metadata_format_supported($metadataPrefix)) )
             {
                $response .="<error code=\"cannotDisseminateFormat\">Metadata Prefix \
                                $metadataPrefix not supported by this archive</error>\n";
             }

        }




# LTRS IMPLEMENTATION USES SETS (does not use this method)
# NACA IMPLEMENTATION DOES NOT USE SETS (use this method)
# NTRS : some internal archives will support sets
       ### NOTE: Sets are not implemented by these repositories, so this error is
implemented
       # ---------------------
       # NO_SET_HIERARCHY CASE (w. ListIdentifiers/ListRecords/ListSets)
       # ---------------------
#       if ( $set )
#               { $response .= "<error code=\"noSetHierarchy\">(Set) argument is not
supported \
#                               by this archive</error>\n";
#       }





        # ---------------------
        # IDENTIFY specific errors
        # ---------------------
        if (  ($verb eq "Identify") && ($count>1) )
                {
&arg_error_check("metadataPrefix","from","until","set","identifier","resumptionToken");
                }



        # LIST_IDENTIFIERS and LIST_RECORDS specific errors
        # ----------------------
        if (  (($verb eq "ListIdentifiers") || ($verb eq "ListRecords"))
&&!($resumptionToken)  )
          {
            #BAD_ARGUMENT
                &arg_error_check("identifier");
                &required_arg_check("metadataPrefix");
          }


        # LIST_METADATAFORMATS CASE
        # ----------------------
        if ($verb eq "ListMetadataFormats" )
          {
            #BAD_ARGUMENT

&arg_error_check("metadataPrefix","from","until","set","resumptionToken");

            #ID_DOES_NOT_EXIST CHECK ( id is an optional argument )
            if ($identifier) {
                &id_check($identifier);
            }
          }


        # LIST_SETS CASE
        # ----------------------
        if ($verb eq "ListSets" )
          {
#           $response .= "<error code=\"noSetHierarchy\">This repository does not support
sets</error>\n";

            #BAD_ARGUMENT
             &arg_error_check("metadadaPrefix","from","until","set","identifier");

             if ($resumption) {
                $response .= "<error code=badArgument>Status: 400 resumptionToken not \
                              supported by this archive with ListSets verb</error>\n";
                }

          }


        # GET_RECORD CASE
        # ----------------------
        if ($verb eq "GetRecord" )
          {
            #BAD_ARGUMENT CHECK
                &arg_error_check("from","until","set","resumptionToken");
                &required_arg_check("metadataPrefix","identifier");
            #ID_DOES_NOT_EXIST CHECK
                &id_check($identifier);

             if ($resumption) {
                $response .= "<error code=badArgument>Status: 400 resumptionToken not \
                              supported by this archive with GetRecord verb</error>\n";
                }
          }

   return ("$response");
}


########################################################################
# Function:     id_check
# Use:          appends <error> to $response if record for $identifier does not exist
# Created:      06/19/2002 TLH
########################################################################
sub id_check {
        my($the_id) = @_;
       my($exists) = &id_exists($the_id);

        if (!($exists)) {
          { $response .="<error code=\"idDoesNotExist\">ID $archive   \"$the_id\" does
not \
                       exist response=  $response (should be 0) </error>\n"; }
        }

}

######################################################################
# Function:     arg_error_check
# Use:          appends <error> to $response if any arg passed in is present
#               This method is called to search verbs calls for inappropriate use of args
# Created:      06/17/2002 TLH
#######################################################################

sub arg_error_check {
        my(@the_args)= @_;
        foreach $the_arg (@the_args)
        {
          if (${$the_arg})
          { $response .="<error code=\"badArgument\">$verb does not permit \"$the_arg\"
argument</error>\n"; }
        }
}




#######################################################################
# Function:     required_arg_check
# Use:          appends <error> to $response if the required args are not passed in
#               This method is called to search verbs calls for required args
# Created:      06/19/2002 TLH
#######################################################################

sub required_arg_check {
        my(@the_args)= @_;
        foreach $the_arg (@the_args)
        {
          if (! ${$the_arg} )
          { $response .="<error code=\"badArgument\">$verb requires use \
                        of the \"$the_arg\" argument</error>\n"; }
        }
}

#####################################################################
# Function:     set_membership_of_record
# Use:          return the set to which a records belongs, given that record
#              Used in such verbs as ListRecords
# Created:      06/18/2002 TLH
# How:          The Dublin Core metadata for a record is passed in as a string.
#                  It (now called $metadata) undergoes pattern matching for the
#                  existence of the "subject" element.
#                  The match ($&) is stored as $setSpec (for readability) and returned
#                  along with the appropriate <setSpec> element tags.
#                  If no "subjects" were found, "" is returned
#
# Modified:     07/16/2002 TLH to work w. <dc:subject> and <subject>, taking into
#                  account greedy pattern matching for handling mutiple subjects.
#                  It is assumed that subject data falls on consecutive lines of the
#                  dc record
# Modified:    07/19/2002 TLH to accept reporitory names as set values as well
#              - ie : ltrs.larc.nasa.gov
######################################################################

sub set_membership_of_record {
        my($id) = @_;
        my ($response,$subject);

       # setSpec already computed in &get_ids and placed in %full
       if ($full{$id}{'shortName'}) {
              $response .="<setSpec>$full{$id}{'shortName'}</setSpec>\n";
       }

        ($subject) = split(' ',$full{$id}{'subject'});
       if ( ($nasa_sti_subjects{$subject}) && ($id =~ /nasa.gov/) ) {
              $response .="<setSpec>$subject</setSpec>\n";
       }

       return ("$response");
}
```

```
#######################################################################
# Function:     set_in_metadata
# Use:          Returns true if the record passed in belongs to the set passed in
# Created:      06/18/2002 TLH
# Note:         "set" maps to the "subject" element of dc metadata of a record
# Modified     07/17/2002 TLH
# Updated:     05/13/2003 MLN - we now pull the archive name from id; no
#                             longer use metadata
#######################################################################

sub set_in_metadata {
        my($set,$id) = @_;
       my($subject);

       # set = archive...
        (undef,$archive) = split(":", $id);
        if ($set eq $full{$id}{'shortName'}) {
                return (1);
        }

       # set = subject...
       ($subject) = split(' ',$full{$id}{'subject'});
        if  ( ($set eq $subject) && ($id =~ /nasa.gov/) ) {
                return (1);
        }

       return(0);
}

##################################################################
# Function:    check_date_format
#              returns true, if arg matches granularity condition (YYYY-MM_DD)
# Created:     06/20/2002 TLH
# Use:         Will try to find the YYYY-MM-DD in the arg
########################################################################

 sub check_date_format {
        my($arg) = @_;
       my($date)=${$arg};
        my($match);

        if ($date =~ /^\d\d\d\d-\d\d-\d\d$/)
       {
               # Check to see if dates are reasonable....

               ($year,$month,$day) = split(/-/, "$date");
               if  ( (! $year =~ /\d\d\d\d/) ||   ($month < 1) || ($month > 12) || ($day
< 1) || ($day > 31) )
               { return (0); }

               # If it gets here, it must be OK !
               return (1);
       }
       else
       { return (0); }
}

########################################################################
# Function:     get_response_date
# Use:          returns a "Complete date" variant of ISO8601
# Created:      11/03/2000 MLN
########################################################################

sub get_response_date {
       my($seconds) = @_;
       my ($time);

       if ($seconds) {
               ($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst) =
gmtime($seconds);
       } else {
               ($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst) = gmtime;
       }

       # now format the response

        if ($year < 100) {
                $year = "19$year";
        } elsif ( ($year >=100) && ($year <2000) ) {
                $year += 1900;
        }
        $mon++;         # 0 indexed
        if ($mon < 10) { $mon = "0$mon"; }
         if ($mday < 10) { $mday = "0$mday"; }
        if ($hour < 10) { $hour = "0$hour"; }
        if ($min < 10) { $min = "0$min"; }
        if ($sec < 10) { $sec = "0$sec"; }

        $time = "$year-$mon-$mday" . "T" . "$hour:$min:$sec+00:00";
        return ("$time");

}

########################################################################
# Function:     metadata_format_supported
# Use:          returns 1 if format is supported, 0 if not
# Created:      11/09/2000 MLN
########################################################################

sub metadata_format_supported {
        my ($format) = @_;
        my (@formats) = ("oai_dc");

        foreach $f (@formats) {
                if ($format eq $f) {
                        return (1);
                }
        }
        return (0);
}

########################################################################
# Function:     get_metadata_formats
# Use:          returns all metadata formats supported by this archive
# Created:      11/03/2000 MLN
# Modified:    07/21/2002 TLH redid schema locations
########################################################################

sub get_metadata_formats {
        my($id) = @_;
        my($response);

        # currently only DC is supported

        $response .= "<metadataFormat>\n";
        $response .= "<metadataPrefix>oai_dc</metadataPrefix>\n";
         $response .= "<schema>http://www.openarchives.org/OAI/2.0/oai_dc.xsd</schema>\n";
         $response
.="<metadataNamespace>http://www.openarchives.org/OAI/2.0/oai_dc/</metadataNamespace>\n";
        $response .= "</metadataFormat>\n";

        return ($response);
}

########################################################################
# Function:     get_sets
# Use:          returns all sets in NTRS DL
#               sets currently used
# Created:      11/03/2000 MLN
# Changed:      01/14/2001 MLN
# Changed:      06/28/2002 TLH
# Updated:     07/16/2002 TLH Added to NTRS implementation
# Updated:     05/13/2003 MLN - converted to MySQL acccess
########################################################################

sub get_sets {
        my($dbh,$sth,$sql,$shortName,$longName);
        my(@sets,$set);

        # find archive based sets...
         $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
                 || db_error("$DBI::errstr");
        $sql = "SELECT shortName, longName from archives";
        $sth = $dbh->prepare($sql);
        $sth->execute;
        while ( ($shortName,$longName) = $sth->fetchrow_array ) {
                push(@sets,"<setSpec>$shortName</setSpec>\n<setName>$longName</setName>");
        }
        $dbh->disconnect;

        # find subject based sets...
        foreach $set (sort(keys(%nasa_sti_subjects))) {
```

```
        push(@sets,"<setSpec>$set</setSpec>\n<setName>$nasa_sti_subjects{$set}</setName>")
; }

        return(@sets);
}

########################################################################
# Function:    get_ids
# Use:         returns all ids in NTRS DL
#              no support for "deleted" ids (yet?)
# Created:     11/03/2000 MLN
# Updated:     02/01/2002 MLN -- handles resumptionTokens
# Updated:     07/11/2002 TLH -- converted for NTRS use (multi repositories)
# Updated:     05/13/2003 MLN - converted to MySQL acccess
########################################################

sub get_ids {
        my($dbh,$sth,$sql,$id,$date,$subject,$shortName,@ids);

        # we have a cache...
        if (-f $id_db) {
                open(F,"$id_db") || &complain("cannot open $id_db - $!\n");
                while (<F>) {
                        ($id,$date,$shortName,$subject) =
                                split(':::',$_);
                        push(@ids,$id);
                        $full{$id}{'dateStamp'} = $date;
                        $full{$id}{'shortName'} = $shortName;
                        $full{$id}{'subject'} = $subject;
                }
                close(F);
                return(@ids);
        }

        # cache not present; hit the database and write the cache

        $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
                 || db_error("$DBI::errstr");

        $sql = "SELECT oaiID, dateStamp, subject, shortName FROM \
                metadata, archives WHERE  \
                ( (metadata.archiveID = archives.archiveID) ) \
                ORDER BY dateStamp ASC";

        $sth = $dbh->prepare($sql);
        $sth->execute;
        open(F,">$id_db.$$") || &complain("cannot open $id_db - $!\n");
        while ( ($id,$date,$subject,$shortName) = $sth->fetchrow_array ) {
                # some subjects can have line feeds; that's bad
                $subject =~ s/\n/ /g;
                push(@ids,$id);
                $full{$id}{'dateStamp'} = $date;
                $full{$id}{'shortName'} = $shortName;
                $full{$id}{'subject'} = $subject;
                print F "$id".":::"."$date".":::"."$shortName".":::"."$subject\n";
        }

        close(F);
        rename("$id_db.$$","$id_db");
        $dbh->disconnect;

        return (@ids);   #returns all ids
}

########################################################################
# Function:     identify
# Use:          returns the archive description
#               could be more complicated in the future
# Created:      11/03/2000 MLN
# Updated:      06/17/2002 TLH  Converted to OAI 2.0 Protocol
########################################################################

sub identify {

        my ($response);

        $response = "<repositoryName>$dl</repositoryName>\n";
        $response .= "<baseURL>http://ntrs.nasa.gov/</baseURL>\n";
        $response .= "<protocolVersion>$oai_version</protocolVersion>\n";
        $response .= "<adminEmail>m.l.nelson\@larc.nasa.gov</adminEmail>\n";
        $response .= "<adminEmail>joanne.rocker\@nasa.gov</adminEmail>\n";
        $response .= "<earliestDatestamp>$earliest_datestamp</earliestDatestamp>\n";
        $response .= "<deletedRecord>$deleted_record</deletedRecord>\n";
        $response .= "<granularity>$granularity</granularity>\n";

        # defining <description> here...
        $response .=    "<description>\n" .
                        "  <oai-identifier \n" .
                        "    xmlns=\"http://www.openarchives.org/OAI/2.0/oai-identifier\"
\n" .
                        "    xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n"
.
                        "    xsi:schemaLocation=  \n" .
                        "       \"http://www.openarchives.org/OAI/2.0/oai-identifier \n "
.
                        "         http://www.openarchives.org/OAI/2.0/oai-
identifier.xsd\">  \n" .
                        "  <scheme>oai</scheme> \n" .
                        "  <repositoryIdentifier>$repositoryId</repositoryIdentifier> \n"
.
                        "  <delimiter>:</delimiter> \n" .
                        "  <sampleIdentifier>oai:naca.larc.nasa.gov:1958:naca-tn-
4410</sampleIdentifier>\n" .
                        " </oai-identifier> \n" .
                        "</description> \n";

        # defining <friends> here...
        $response .= "<description>\n" .
        "  <friends \n" .
        "     xmlns=\"http://www.openarchives.org/OAI/2.0/friends/\" \n" .
        "     xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n" .
        "     xsi:schemaLocation= \n" .
        "          \"http://www.openarchives.org/OAI/2.0/friends/ \n" .
        "            http://www.openarchives.org/OAI/2.0/friends.xsd\"> \n" .
        "     <baseURL>http://techreports.larc.nasa.gov/ltrs/oai2.0/</baseURL>\n" .
       "     <baseURL>http://naca.central.cranfield.ac.uk/cgi-bin/nph-
oai.cgi</baseURL>\n" .
       "     <baseURL>http://ston.jsc.nasa.gov/collections/TRS/oai/</baseURL>\n" .
       "     <baseURL>http://trs.nis.nasa.gov/perl/oai/</baseURL>\n" .
       "     <baseURL>http://naca.larc.nasa.gov/oai2.0/</baseURL>\n" .
       "     <baseURL>http://eprints.riacs.edu/perl/oai/</baseURL>\n" .
       "     <baseURL>http://celestial.eprints.org/cgi-bin/oaia2/arXiv.org</baseURL>\n" .
       "     <baseURL>http://www.biomedcentral.com/oai/2.0/</baseURL>\n" .
       "     <baseURL>http://www-dev.osti.gov/oai2.0/</baseURL>\n" .
       "     <baseURL>http://genesis2.jpl.nasa.gov/perl/oai</baseURL>\n" .
       "     <baseURL>http://www.giss.nasa.gov/cgi-bin/gpol2/</baseURL>\n" .
        "  </friends> \n" .
        "</description> \n";

        return ($response);
}

########################################################################
# Function:     get_datestamp
# Use:          returns a datestamp in the form of:
#              YYYY-MM-DD
#              computed from the metadata file
# Created:      11/03/2000 MLN
# Modified:    07/11/2002 TLH for NTRS implementation:
# Modified:    05/15/2003 MLN simplified
########################################################################

sub get_datestamp {

       my($id) = @_;
       my($dateStamp);

       if ($full{$id}{'dateStamp'}) {
               # this is a LR/LI, and the cache has been read in
               return ($full{$id}{'dateStamp'});
       } else {
               # this is a GR, and its quicker to hit MySQL rather than
               # read in the whole cache

               $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
                       || db_error("$DBI::errstr");
               $sql = "SELECT dateStamp FROM metadata WHERE oaiID='$id'";
               $sth = $dbh->prepare($sql);
               $sth->execute;
               ($dateStamp) = $sth->fetchrow_array;
               $sth->finish;
```

```
                $dbh->disconnect;
                return($dateStamp);
        }
}

########################################################################
# Function:     split_id
# Use:          returns the year and report # from an id
# Created:      11/03/2000 MLN
# Updated:     07/11/2002 TLH
########################################################################

sub split_id {

        my($id) = @_;

        ($oai,$archive,$year,$report) = split(":",$id);

        return;

}


########################################################################
# Function:     generate_empty_DC
# Use:          returns xml formatted DC repsonse for valid id that
#               contains no metadata (empty record)
# Created:      06/28/2002 TLH
########################################################################
 sub generate_empty_DC {

        my($stuff) = @_;
         $response = "<oai_dc:dc \n";
         $response .="      xmlns:oai_dc=\"http://www.openarchives.org/OAI/2.0/oai_dc/\"
\n";
         $response .="      xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n";
         $response .="      xmlns:dc=\"http://purl.org/dc/elements/1.1/\" \n";
         $response .="
xsi:schemaLocation=\"http://www.openarchives.org/OAI/2.0/oai_dc/ \n";
         $response .="      http://www.openarchives.org/OAI/2.0/oai_dc.xsd\"> \n";

         $response .= "<dc:description>NO METADATA AVAILABLE $stuff</dc:description> \n";
         $response .= "</oai_dc:dc> \n";

         return $response;
}



########################################################################
# Function:     get_dc
# Use:          returns the metadata in DC format given an id
#              currently, the metadata is in metadata directory
# Created:      11/03/2000 MLN
# Updated:      12/28/2001 MLN - no longer complains if file won't open
# Updated:     07/09/2002 TLH - converted for NTRS use
# Updated:     05/13/2003 MLN - converted to MySQL acccess
# Updated:     05/19/2003 MLN - handles provenance
########################################################################

sub get_dc {
         my($id) = @_;
         my($metadata,$archive,$provenance);

        # we're assuming dc format here...

         $response = "<metadata>\n<oai_dc:dc ";
         $response .="xmlns:oai_dc=\"http://www.openarchives.org/OAI/2.0/oai_dc/\" \n";
         $response .="xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\" \n";
         $response .="xmlns:dc=\"http://purl.org/dc/elements/1.1/\" \n";
         $response .="xsi:schemaLocation=\"http://www.openarchives.org/OAI/2.0/oai_dc/
\n";
         $response .="http://www.openarchives.org/OAI/2.0/oai_dc.xsd\"> \n";

         (undef,$archive) = split(":",$id);

        open(F,"$report_root$archive/metadata/$id") ||
            die ("cannot open $report_root/$archive/metadata/$id : $!\n");

        while (<F>) {
                $line = $_;
                # skip the DC lines; the OAI code will put the right stuff in
                next if ($line =~ /<dc .*/);
                next if ($line =~ /<\/dc>/);
                # skip blank lines
                next if ($line =~ /^\s*$/);
                # the next line is a hack to get rid of:
                # http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
                # which should not be confused as part, for example,
                # an abstract
                next if ($line =~ /\s*http:..www.openarchives.org.OAI.*dc.xsd">/);


                 # process provenance records separately
                # the VT harvester puts provenance on 1 line
                 if ($line =~ /<provenance/) {
                        $provenance .= $line;
                        next;
                }

                # process the DC tags correctly
                if ($line =~ /<dc:/) {
                        # case:   <dc:foo>
                        $response .= $line;
                } elsif ($line =~ /<\/dc:/) {
                        # case:   <dc:foo>
                        $response .= $line;
                } elsif ($line =~ /<\w+/) {
                        # case:   <foo>bar</foo>
                        $line =~ s#<#<dc:#;
                        $line =~ s#</#</dc:#;
                        $response .= $line;
                } elsif ($line =~ /<\/\w+/) {
                        # case:   </foo>
                        $line =~ s#</#</dc:#;
                        $response .= $line;
                } else {
                        # case:   (text with no tags)
                        $response .= $line;
                }
        }

        close(F);

        $response .= "</oai_dc:dc>\n</metadata>\n";

        # add a provenance record; use existing provenance too if it exists
        $response .= &get_provenance($id,$provenance);

        return("$response");

}

########################################################################
#Function:      get_provenance
#Use:           returns provenance record(s) for an id
#Created:       05/19/2003
########################################################################

sub get_provenance {
        my($id,$previous_provenance) = @_;
        my ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst);
        my ($r);

        $provenance_line1 = "<provenance
xmlns=\"http://www.openarchives.org/OAI/2.0/provenance\"
xmlns:xsi=\"http://www.w3.org/2001/XMLSchema-instance\"
xsi:schemaLocation=\"http://www.openarchives.org/OAI/2.0/provenance
http://www.openarchives.org/OAI/2.0/provenance.xsd\">";
        $provenance_line2 = "</provenance>";

        # these archives are not real OAI-PMH archives, thus they have no
        # real provenance (wrt to NTRS); so return nothing

        if ( ($id =~ /casi.ntrs.nasa.gov/) || ($id =~ /atrs.arc.nasa.gov/) ||
             ($id =~ /ssctrs.ssc.nasa.gov/) || ($id =~ /gtrs.gsfc.nasa.gov/)
               || ($id =~ /ktrs.ksc.nasa.gov/) )
        {
                return("");
        }

        # these records will have at least 1 provenance to fill out
        # we'll get some of the values from the metadata file, and some
        # from the $archive/data/config.xml file used by the VT harvester

        # NTRS currently doesn't change things...
        $altered = "false";

        # harvestDate
        (undef,$archive) = split(':',$id);
         $seconds = (stat("$report_root$archive/metadata/$id")) [9];
        ($sec,$min,$hour,$mday,$mon,$year,$wday,$yday,$isdst) = localtime($seconds);

        $harvestDate = &get_response_date($seconds);

        # baseURL
        open(B,"$report_root$archive/config.xml") || &complain("cannot open
$report_root$archive/config.xml - $!\n");

        while (<B>) {
                next unless (/<url>/);
                $baseURL = $_;
                $baseURL =~ s#</*url>##g;
                $baseURL =~ s#\s*##g;
                close(B);
        }

        # id is unchanged
        $id = $id;

        # datestamp
        $datestamp = &get_datestamp($id);

        # metadataNamespace
        $metadataNamespace = "http://www.openarchives.org/OAI/2.0/oai_dc/";


        $r = "<about>$provenance_line1\n";
        $r .= "<originDescription harvestDate=\"$harvestDate\" altered=\"$altered\">\n";
        $r .= "<baseURL>$baseURL</baseURL>\n";
        $r .= "<identifier>$id</identifier>\n";
        $r .= "<datestamp>$datestamp</datestamp>\n";
        $r .= "<metadataNamespace>$metadataNamespace</metadataNamespace>\n";
        if ($previous_provenance) {
                # this id already has a provenance; we must nest the existing
                # provenance inside ours
                # strip off the any old <provenance .*.xsd"> tags
                $previous_provenance =~ s#<provenance.*\.xsd">##;
                $previous_provenance =~ s#</provenance>##;
                $r .= $previous_provenance;
        }
        $r .= "</originDescription>\n$provenance_line2\n</about>\n";

        return($r);

}

#########################################################################
#Function:      parse_for_element
#Use:           extracts occurances of a given element from a given xml
#               instance (passed in).  This need for this function arose
#               from metadata containing multiple instances of certain
#               elements, which are non-consecutive.
#               ie:  <dc:type>foo<dc:/type><dc:date>bar</dc:date><dc:type...
#Sample Call:   parse_for_element("dc:type","$data");
#Author:        TLH
#Created:       07/18/2002
#Updated:       07/23/2002 TLH added "is" to pattern match
#########################################################################

sub parse_for_element {

        my($dc_element, $data) = @_;
        my($result) = "";

        @blocks = split ("<$dc_element>",$data);
          #get rid of 1st block w. shift...
        shift(@blocks);

         foreach $block (@blocks)
         {

                        # ADD START TAG BACK
                 $block   = "<$dc_element>$block";  # </$dc_element>";
                        # FIND JUST THE INSTANCE OF IT...
                  $block   =~ /<$dc_element>.*<\/$dc_element>/is ;
                  $block   = $&;
                        # GET RID OF TAGS
                $block   =~ s#<$dc_element>##;
                $block   =~ s#</$dc_element>##;
                        # REMOVE ENTITIES FROM DATA...
                $block   = &remove_entities($block);
                        # ADD TAGS BACK..
                $block  = "<$dc_element>$block</$dc_element>";

                  $result .= "$block \n";
         }

         return "$result" ;
}

########################################################################
#Function:     parse_out_all_but
#Use:          - Removes all dc elements (and their content), except for the
#                element (and content of) passed in,  from the string passed in
#                (typically the result of a greedy pattern match)
#Created:      TLH
########################################################################

#sub parse_out_all_but {
#       my($element, $data) = @_;
#
#       if ($element ne "title")
#               { $data  =~ s/<dc:title>.*<\/dc:title>//gi; }
#
#        elsif ($element ne "creator")
#                 { $data  =~ s/<dc:creator>.*<\/dc:creator>//gi; }
#
#        if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#         if ($element ne "subject")
#                 { $data  =~ s/<dc:subject>.*<\/dc:subject>//gi; }
#
#}
```

```
########################################################################
# Function:    remove_entities
# Use:         replaces special chars w. their pre-defined entities (xml specific set)
# Created:     MLN
# Updated:     7.15.2002 TLH
########################################################################


sub remove_entities {
        my($string) = @_;

        %entities = (
                "&"   =>      "amp",
                ">"   =>      "gt",
                "<"   =>      "lt",
                "\'"  =>      "apos",
                "\""  =>      "quot"
        );


        foreach $e (keys(%entities)) {
                $string =~ s/$e/\&$entities{"$e"};/g;
        }

        return($string);

}
```

```
########################################################################
# Function:     encode_xml
# Use:          takes a string as input and returns that same string
#               with all of its unicode character references encoded
# Note:        having to do this is stupid.
# Created:      01/12/2001 MLN
########################################################################


sub encode_xml {
       my($input) = @_;

       $input =~ s#\&#\&amp;#g;
       $input =~ s#<#\&lt;#g;
       $input =~ s#>#\&gt;#g;

       return($input);

}

########################################################################
# Function:     encode_url
# Use:          takes a string as input and returns that same string
#              with all of its special http characters encoded
# Created:      11/08/2000 MLN
########################################################################


sub encode_url {

       my($input) = @_;

       $input =~ s#\/#%2F#g;
       $input =~ s#\?#%3F#g;
       $input =~ s/\#/%23/g;
       $input =~ s#\=#%3D#g;
       $input =~ s#\&#%26#g;
       $input =~ s#\:#%3A#g;
       $input =~ s#\;#%3B#g;
       $input =~ s#\+#%20#g;

       return($input);
}
```

```
########################################################################
# Function:     strip_chars
# Use:          strips non-printable chars from the text
# Created:      12/28/2001 MLN
########################################################################

sub strip_chars {
        my ($line) = @_;
        my (@chars, $c, @newchars, $newline);

        @chars = split(//, $line);

        foreach $c (@chars) {
                next if ( (ord($c) > 127) || (ord($c) == 11) );
                push (@newchars, $c);
        }

        $newline = join('',@newchars);

        return($newline);
}

########################################################################
# Function:     get_next_resumption
# Use:          returns the next resumption token, or nothing if done
# Created:      02/01/2002 MLN
########################################################################

sub get_next_resumption {
       my($resumption) = @_;
       my($from,$until,$metadataPrefix,$set,$start_range,$end_range);

       ($from,$until,$metadataPrefix,$set,$start_range,$end_range) =
               &decode_resumption($resumption);

       $end_range = $start_range + ($max_records_to_return * 2) - 1;
       $start_range += $max_records_to_return;

       $resumption = &encode_resumption($from,$until,$metadataPrefix,
                       $set,$start_range,$end_range);

       return($resumption);
}

########################################################################
# Function:     decode_resumption
# Use:          decodes a resumptionToken
# Created:      02/15/2002 MLN
########################################################################

sub decode_resumption {
        my($resumption) = @_;
       my($from,$until,$metadataPrefix,$set,$start_range,$end_range);

       ($from,$until,$metadataPrefix,$set,$start_range,$end_range) =
               split(/!/,$resumption);

       return($from,$until,$metadataPrefix,$set,$start_range,$end_range);
}

########################################################################
# Function:     encode_resumption
# Use:          encodes a resumptionToken
# Created:      02/15/2002 MLN
########################################################################

sub encode_resumption {
        my($from,$until,$metadataPrefix,$set,$start_range,$end_range) = @_;
        my($resumption);

       $resumption = join("!",$from,$until,$metadataPrefix,$set,
               $start_range,$end_range);
       return($resumption);
}

########################################################################
# Function:     id_exists
# Use:          returns "0" if id does not exist, 1 if it does
# Created:      11/09/2000 MLN
# Modified:    07/09/2002 TLH  for NTRS implementation
# Modified:    05/15/2003 MLN  this used to check if $full{$id} exists
#              but for efficiency, we no longer build %full for
#              single GetRecord requests; we test for the file
#########################################################################

sub id_exists {
       my($id) = @_;
       my($archive);

        (undef,$archive) = split(":",$id);

       if (-f "$report_root$archive/metadata/$id")
          { return (1); }
       else
          { return (0); }
}

#########################################################################
# Function:     sys_load
# Use:          returns the system load for the last minute
#              essentially the first average reported in "uptime"
#              it would be nice to find the load with something more
#              portable than "uptime"
# Created:      01/11/2001 MLN
#########################################################################

sub sys_load {
       my ($uptime, @uptime, $minute_load);

       # uptime output looks like:
       # 4:16pm  up 3 days, 25 min, 12 users,  load average: 0.00, 0.05, 0.07
       # use the 11th column and remove the trailing ","

       $uptime = uptime();
       @uptime = split(' ',$uptime);
       $minute_load = $uptime[10];
       $minute_load =~ s/,//g;

       return($minute_load);
}

#########################################################################
# Function:     too_busy
# Use:          redirects the request to another server if this one
#              is too busy.  ListIdentifiers, ListRecords and ListSets
#              can cause this to happen
# Created:      01/11/2001 MLN
#########################################################################

sub too_busy {

       $load = &sys_load;

       $threshold = &internal_read_state("threshold");
       $oai_backup = &internal_read_state("oai_backup");

       return if ( ($oai_backup eq "null") || !($oai_backup) );

       if ("$load" > "$threshold") {
              $backup = "$oai_backup" . "?" . $ENV{"QUERY_STRING"};
              &log("too_busy","OK","redirecting to $backup");
              print "Location: $backup\n\n";
              exit;   # don't keep on processing
       }

}

1;
::::::::::::::
oai/sets.pl
::::::::::::::
%nasa_sti_subjects = (
      "01" => "AERONAUTICS",
      "02" => "AERODYNAMICS",
      "03" => "AIR TRANSPORTATION AND SAFETY",
      "04" => "AIRCRAFT COMMUNICATIONS AND NAVIGATIONS",
      "05" => "AIRCRAFT DESIGN, TESTING AND PERFORMANCE",
      "06" => "AVIONICS AND AIRCRAFT INSTRUMENTATION",
      "07" => "AIRCRAFT PROPULSION AND POWER",
      "08" => "AIRCRAFT STABILITY AND CONTROL",
      "09" => "RESEARCH AND SUPPORT FACILITIES (AIR)",
      "12" => "ASTRONAUTICS ",
      "13" => "ASTRODYNAMICS",
      "14" => "GROUND SUPPORT SYSTEMS AND FACILITIES (SPACE)",
      "15" => "LAUNCH VEHICLES AND LAUNCH OPERATIONS",
      "16" => "SPACE TRANSPORTATION AND SAFETY",
      "17" => "SPACE COMMUNICATIONS, SPACECRAFT COMMUNICATIONS,",
      "18" => "SPACECRAFT DESIGN, TESTING AND PERFORMANCE",
      "19" => "SPACECRAFT INSTRUMENTATION AND ASTRIONICS",
      "20" => "SPACECRAFT PROPULSION AND POWER",
      "23" => "CHEMISTRY AND MATERIALS ",
      "24" => "COMPOSITE MATERIALS",
      "25" => "INORGANIC, ORGANIC AND PHYSICAL CHEMISTRY",
      "26" => "METALS AND METALLIC MATERIALS",
      "27" => "NONMETALLIC MATERIALS",
      "28" => "PROPELLANTS AND FUELS",
      "29" => "SPACE PROCESSING",
      "31" => "ENGINEERING ",
      "32" => "COMMUNICATIONS AND RADAR",
      "33" => "ELECTRONICS AND ELECTRICAL ENGINEERING",
      "34" => "FLUID MECHANICS AND THERMODYNAMICS",
      "35" => "INSTRUMENTATION AND PHOTOGRAPHY",
      "36" => "LASERS AND MASERS",
      "37" => "MECHANICAL ENGINEERING",
      "38" => "QUALITY ASSURANCE AND RELIABILITY",
      "39" => "STRUCTURAL MECHANICS",
      "42" => "GEOSCIENCES ",
      "43" => "EARTH RESOURCES AND REMOTE SENSING",
      "44" => "ENERGY PRODUCTION AND CONVERSION",
      "45" => "ENVIRONMENT POLLUTION",
      "46" => "GEOPHYSICS",
      "47" => "METEOROLOGY AND CLIMATOLOGY",
      "48" => "OCEANOGRAPHY",
      "51" => "LIFE SCIENCES ",
      "52" => "AEROSPACE MEDICINE",
      "53" => "BEHAVIORAL SCIENCES",
      "54" => "MAN/SYSTEMS TECHNOLOGY AND LIFE SUPPORT",
      "55" => "EXOBIOLOGY",
      "59" => "MATHEMATICAL AND COMPUTER SCIENCES ",
      "60" => "COMPUTER OPERATIONS AND HARDWARE",
      "61" => "COMPUTER PROGRAMMING AND SOFTWARE",
      "62" => "COMPUTER SYSTEMS",
      "63" => "CYBERNETICS, ARTIFICIAL INTELLIGENCE AND ROBOTICS",
      "64" => "NUMERICAL ANALYSIS",
      "65" => "STATISTICS AND PROBABILITY",
      "66" => "SYSTEMS ANALYSIS AND OPERATIONS RESEARCH",
      "67" => "THEORETICAL MATHEMATICS",
      "70" => "PHYSICS ",
      "71" => "ACOUSTICS",
      "72" => "ATOMIC AND MOLECULAR PHYSICS",
      "73" => "NUCLEAR PHYSICS",
      "74" => "OPTICS",
      "75" => "PLASMA PHYSICS",
      "76" => "SOLID-STATE PHYSICS",
      "77" => "PHYSICS OF ELEMENTARY PARTICLES AND FIELDS",
      "80" => "SOCIAL AND INFORMATION SCIENCES ",
      "81" => "ADMINISTRATION AND MANAGEMENT",
      "82" => "DOCUMENTATION AND INFORMATION SCIENCE",
      "83" => "ECONOMICS AND COST ANALYSIS",
      "84" => "LAW, POLITICAL SCIENCE AND SPACE POLICY",
      "85" => "TECHNOLOGY UTILIZATION AND SURFACE TRANSPORTATION",
      "88" => "SPACE SCIENCES ",
      "89" => "ASTRONOMY",
      "90" => "ASTROPHYSICS",
      "91" => "LUNAR AND PLANETARY SCIENCE AND EXPLORATION",
      "92" => "SOLAR PHYSICS",
      "93" => "SPACE RADIATION",
      "99" => "GENERAL",
);

1;
```

Appendix 4: NTRS Methods Source Code

```
::::::::::::::
about.pl
::::::::::::::
########################################################################
# Method:       about
# Use:                  about page for NTRS
# Created:      05/07/2002 MLN
########################################################################

sub about
{
        &http_header("text/html");
        my($base) = &MyBaseUrl;

        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
               # turn on the "about search" button
               if (/<!--about-->/) {
                      s/dedede/white/g;
               }
                $header .= $_;
        }
        close(H);

        print "$header\n";

        open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
        while (<F>) {
                $footer .= $_;
        }
        close(F);

        print <<EOF;

<h2>
NASA STI Program and NTRS
</h2>
</center>

<blockquote>
The <a href="http://www.sti.nasa.gov">NASA Scientific and Technical Information
(STI) Program's </a> mission is to collect, archive, and disseminate NASA
aerospace information, and locate domestic and international STI pertinent
to NASA's missions and Strategic Enterprises.  Examples of NASA's STI
include research reports, journal articles, conference and meeting papers,
technical videos, mission-related operational documents, and preliminary
data. NASA's technical information is available via the NASA Technical
Report Server (NTRS) is to provide students, educators, and the public
access to NASA's technical literature. NTRS also collects scientific and
technical information from sites external to NASA to broaden the scope of
information available to users. NTRS's <a href="?method=simple">Simple
Search</a> searches for NASA information only and its <a
href="?method=advanced">Advanced Search</a> can search for NASA and
non-NASA information. Most of the NASA information does not have full-text
document images and documents can be ordered by contacting the <a
href="?method=ordering"> NASA Center for AeroSpace Information</a>.

</blockquote>

<center>
<p>
<h2>About the Collections in NTRS</h2>
</center>
<blockquote>

The NTRS collects scientific and technical information from NASA's
technical report servers and non-NASA sites using the <a
href="http://www.openarchives.org/">Open Archives Initiative </a> Protocol
for Metadata Harvesting (OAI-PMH). <b>Information  Disclaimer:</b> The
external sites selected for inclusion in the NTRS are <i>not endorsed or
maintained </i> by NASA. The validity, accuracy, integrity, and maintenance
of information from external sources are the responsibility of the
sponsoring organization. Any questions regarding the information found in
```

```
non-NASA resources should be referred to the appropriate institutional
sponsors.
</blockquote>
<p>
<center>

EOF

        use DBI;

        $datasource = "DBI:mysql:database=ntrs;host=localhost";
        $user = "mln";
        $password = "ntrsr0cks";

        #$dbh = DBI->connect($datasource, $user, $password, {'RaiseError' => 1});
        $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0}) ||
db_error("$DBI::errstr");

        $sql = "SELECT * FROM archives ORDER BY longName";
        $sth = $dbh->prepare($sql);
        $sth->execute;

        print "<table><tr bgcolor=$grey><th>Archive<th>Maintained <br>by NASA<th>Last
<br>Harvest <th>Last <br>Update <th>Records <br>in NTRS\n";
        while ( ($archive_id,$archive_shortname,$archive_longname,
                $archive_nasa,$archive_url,$size) = $sth->fetchrow_array )
        {
                $total += $size;

                if ($archive_nasa eq "nasa") { $nasa = "yes"; }
                        else { $nasa = "no";}

                $date = &last_successful_harvest($archive_shortname);
                $update = &last_new_record($archive_shortname);

                if ($archive_url) {
                        print "<tr><td><a href=$archive_url>$archive_longname</a><td
align=center>$nasa<td align=center>$date<td align=center>$update<td align=right>$size\n";

                } else {
                        print "<tr><td>$archive_longname<td align=center>$nasa<td
align=center>$date<td align=center>$update<td align=right>$size\n";
                }
        }
        print
"<tr><td>&nbsp;<td>&nbsp;<td>&nbsp;<td>&nbsp;<td>&nbsp;<tr><td><i>total</i><td><td><td><t
d align=right><i>$total</i></table>\n";

         $sth->finish;

         $dbh->disconnect;

        print <<EOF;

</center>
<blockquote>
<p>
Some archives in the previous version of NTRS that have not yet been
included in this version of NTRS:
<p>
<ul>
<li> NASA Astrophysics Data System:
<a href="http://adsabs.harvard.edu/abstract_service.html">Astronomy & Astrophysics</a>,
<a href="http://adsabs.harvard.edu/physics_service.html">Physics & Geophysics</a>,
<a href="http://adsabs.harvard.edu/instrumentation_service.html">Space
Instrumentation</a>
<li> <a href="http://www.dfrc.nasa.gov/DTRS/">NASA Dryden Flight Research Center</a>
<li> <a href="http://gltrs.grc.nasa.gov/">NASA Glenn Research Center</a>
<li> <a href="http://techreports.jpl.nasa.gov/">NASA Jet Propulsion Laboratory</a>
</ul>
</blockquote>
<center>
<p>
<h2>History of NTRS</h2>
</center>
<blockquote>
The NTRS has been in service since 1994.  The architecture of NTRS has
changed throughout the years, moving from distributed searching to metadata
harvesting with the <a href="http://www.openarchives.org/">OAI-PMH</a>.
Some papers about NTRS and constituent digital libraries are included below.
```

```
<ul>

<li> Michael L. Nelson, JoAnne Rocker and Terry L. Harrison,
<a href="http://techreports.larc.nasa.gov/ltrs/dublincore/2003/jp/NASA-2003-lht-
mln.html"><b>OAI and NASA
Scientific and Technical Information,</b></a> <i>Library Hi-Tech</i>, 21(2), 2003, pp.
140-150.
<p>

<li> Michael L. Nelson, <A
href="http://techreports.larc.nasa.gov/ltrs/refer/1999/tm/NASA-99-
tm209127.refer.html"><B>A Digital Library for the National
Advisory Committee for Aeronautics,</B></a>
NASA/TM-1999-209127, April 1999.
<p>

<LI> Michael L. Nelson and
Ming-Hokng Maa,
<B><A HREF="http://techreports.larc.nasa.gov/ltrs/refer/papers/NASA-96-ir-
p64/">Optimizing the NASA Technical Report Server,</A></B>
<I>Internet Research: Electronic Network Applications and Policy</I>,
6(1), August 1996, pp. 64-70.
<P>

<LI>Michael L. Nelson,
Gretchen L. Gottlich,
David J. Bianco,
Sharon S. Paulson,
Robert L. Binkley,
Yvonne D. Kellogg,
Chris J. Beaumont,
Robert B. Schmunk,
Michael J. Kurtz,
Alberto Accomazzi and
Omar Syed,
<B><A HREF="http://techreports.larc.nasa.gov/ltrs/refer/papers/NASA-95-ir-p25/NASA-95-ir-
p25.html">The NASA Technical Report Server,</A></B>
<I>Internet Research: Electronic Network Applications and Policy</I>,
5(2), September 1995, pp. 25-36.
<P>

<LI>Michael L. Nelson,
Gretchen L. Gottlich,
David J. Bianco,
Robert L. Binkley,
Yvonne D. Kellogg,
Sharon S. Paulson,
Chris J. Beaumont,
Robert B. Schmunk,
Michael J. Kurtz and
Alberto Accomazzi,
<B><A HREF="http://techreports.larc.nasa.gov/ltrs/refer/1995/aiaa-95-0964.refer.html">The
Widest Practicable Dissemination: The NASA Technical Report Server,</A></B>
<I>Computers in Aerospace 10</I>, AIAA-95-0964, San Antonio TX, March 28-30, 1995.
<P>

<LI>Donna G. Roper,
Mary K. McCaskill,
Scott D. Holland,
Joanne L. Walsh,
Michael L. Nelson,
Susan L. Adkins,
Manjula Y. Ambur and
Bryan A. Campbell,
<B><A HREF="http://techreports.larc.nasa.gov/ltrs/refer/1994/tm109172.refer.html">A
Strategy for Electronic Dissemination of NASA Langley Technical
Publications,</A></B>
NASA TM-109172, December 1994.
<P>

<LI>Michael L. Nelson,
Gretchen L. Gottlich and
David J. Bianco,
<B><A
HREF="http://techreports.larc.nasa.gov/ltrs/refer/1994/tm109162.tex.refer.html">World
Wide Web Implementation of the Langley Technical Report Server,</A></B>
NASA TM-109162, September 1994.
<P>

<LI>Michael L. Nelson and
Gretchen L. Gottlich       ,
<B><A
HREF="http://techreports.larc.nasa.gov/ltrs/refer/1994/rdp4567.tex.refer.html">Electronic
Document Distribution: Design of the Anonymous FTP Langley Technical Report
Server,</A></B>
NASA TM-4567, March 1994.
<P>
</ul>

</blockquote>

<center>
EOF

         print "$footer\n";
        &log("about","OK","-");
}

1;


sub last_successful_harvest {
        my ($archive) = @_;

        $file = "/usr/local/web/htdocs/ntrs/ODL-
Harvest/Harvest/$archive/data/$archive.oai_dc.date";

        open(F,"$file") || return ("n/a");
        while (<F>) {
                # there should only be 1 line
                $seconds = $_;
        }
        close (F);

        ($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst) =
                gmtime($seconds);

        $year += 1900;

        $mon++;         # 0 indexed
        if ($mon < 10) { $mon = "0$mon"; }
        if ($mday < 10) { $mday = "0$mday"; }

        return("$year-$mon-$mday");
}

sub last_new_record {
        my ($archive) = @_;
        my ($dev,$ino,$mode,$nlink,$uid,$gid,$rdev,$size, $atime,$mtime,
$ctime,$blksize,$blocks);

        $file = "/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/$archive/metadata";
        ($dev,$ino,$mode,$nlink,$uid,$gid,$rdev,$size, $atime,$mtime,
         $ctime,$blksize,$blocks) = stat($file);

        ($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst) =
                gmtime($mtime);

        $year += 1900;

        $mon++;         # 0 indexed
        if ($mon < 10) { $mon = "0$mon"; }
        if ($mday < 10) { $mday = "0$mday"; }

        return("$year-$mon-$mday");
}
1;
::::::::::::::
admin.pl
::::::::::::::
# Use:                  admin page for NTRS
# Created:      10/10/2002 MLN
########################################################################

sub admin
{
        #print "Refresh: 3\n";
         &http_header("text/html");
        $run = $in{"run"};
        $arg = $in{"arg"} if ($in{"arg"} =~ /^[\w\:\%\.]+$/);
        $arg2 = $in{"arg2"} if ($in{"arg2"} =~ /^\w+$/);
        $archives = $in{"archives"} if ($in{"archives"} =~ /^[\w\.]+$/);
        my($base) = &MyBaseUrl;

        # commands
        $harvest_all = "../bin/harvest-cron";
        $harvest_single = "../ODL-Harvest/Harvest/";
        $populate = "../bin/populate-ntrs.pl";
        $rebuild = "../bin/build-table.pl";
        $delete = "../bin/delete.pl";
        $archive_html = "../bin/archives-html.pl";

         # get headers and footers

         open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
         while (<H>) {
                # turn on the "admin search" button
                if (/<!--admin-->/) {
                        s/dedede/white/g;
                }
                 $header .= $_;
         }
         close(H);

         print "$header\n";

         if ($run) {
                print "<table bgcolor=dedede> <tr> <td>";
                # run a command

                if ($run eq "harvest") {
                        print "</center><pre>\n";
                        if ($archives =~ /\w+/) {
                                # do a single archive
                                if (-x "$harvest_single$archives/harvest.pl") {
                                        print "beginning $archives harvest\n";
                                        $output =
system("$harvest_single$archives/harvest.pl");
                                        print "ending $archives harvest\n";
                                } else {
                                        print "$archives is not a harvestable archive\n";
                                }
                        } else {
                                # do all the archives
                                $output = system("$harvest_all");
                        }
                        print "</pre><center>\n";
                } elsif ($run eq "populate") {
                        print "</center><pre>\n";
                        select (STDOUT); $|++;
                        $output = system("$populate $archives");
                        print "</pre><center>\n";
                } elsif ($run eq "rebuild") {
                        print "</center><pre>\n";
                        $output = system("$rebuild");
                        print "</pre><center>\n";
                } elsif ($run eq "delete") {
                        print "</center><pre>\n";
                        $output = system("$delete $arg2 $arg");
                        print "</pre><center>\n";
                } elsif ($run eq "archives") {
                        print "</center><pre>\n";
                        $output = system("$archive_html");
                        print "</pre><center>\n";
                } else {
                        print "cannot run the program $run\n";
                }
                print "</table> <p>\n";

                print "<a href=?method=admin>Return to the Adminstrative Page</a>\n";
        } else {
                # print the admin page if no command is given
                open(N,"ntrs.pkg/admin.html") || die ("$!: cannot open admin\n");
                while (<N>) {
                        $admin .= $_;
                }
                close(N);

                open(N,"ntrs.pkg/select.html") || die ("$!: cannot open select\n");
                while (<N>) {
                        $select .= $_;
                }
                close(N);
```

```
		# insert the select box
		$admin =~ s/SELECT/$select/g;
		print $admin;
	}

	open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
	while (<F>) {
		$footer .= $_;
	}
	close(F);

	print "$footer\n";
	&log("admin","OK","-");
}

1;
::::::::::::::
advanced.pl
::::::::::::::
########################################################################
# Method:	advanced
# Use:			advanced page for NTRS
# Created:	05/07/2002 MLN
# Updated:	06/19/2003 MLN - uses the static archives html page
########################################################################

sub advanced
{
	&http_header("text/html");
	my($base) = &MyBaseUrl;

	# get headers and footers

	open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
	while (<H>) {
		# turn on the "advanced search" button
		if (/<!--advanced-->/) {
			s/dedede/white/g;
		}
		$header .= $_;
	}
	close(H);

	open(H,"ntrs.pkg/checkbox.html") || die ("$!: cannot open checkbox file\n");
	while (<H>) {
		$checkbox .= $_;
	}
	close(H);


	open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
	while (<F>) {
		$footer .= $_;
	}
	close(F);

	print "$header\n";

	print <<EOF;

<form action=$base method=GET>
<input type=hidden name=method value=search>
<input type=hidden name=offset value=0>
<input type=hidden name=mode value=advanced>
<table>
<tr>
<td align=right width=30%> <label for=title>Title: </label>
<td><input name=title size=40 id=title>
<tr>
<td align=right width=30%> <label for=creator>Author(s): </label>
<td><input name=creator size=40 id=creator>
<tr>
<td align=right width=30%> <label for=date>Date: </label>
<td><input name=date size=40 id=date>
<tr>
<td align=right width=30%> <label for=type>Report Number / <br>Journal / Conference:
</label>
<td><input name=type size=40 id=type>
<tr>
<td align=right width=30%> <label for=description>Abstract: </label>
```

```
<td><input name=description size=40 id=description>

<tr>
<td align=right width=30%> <label for=boolean>Combine fields with:</label>
<td> <input type=radio name=boolean value=and checked id=boolean>and
<input type=radio name=boolean value=or id=boolean>or

<tr>

<td align=right width=30%> <label for=orderby>Order results by:</label>
<td> <select name=orderby id=orderby>
<option selected value=date>Publication date
<option value=dateStamp>Date added to NTRS
<option value=title>Title
<option value=creator>Author(s)
<option value=type>Citation
</select>

<tr>
<td align=right width=30%> <label for=order>Order of results:</label>
<td> <select name=order id=order>
<option selected value=DESC>Descending
<option value=ASC>Ascending
</select>

<tr>
<td align=right width=30%> <label for=limit>Results per page:</label>
<td> <select name=limit id=limit>
<option value=10>10
<option selected value=25>25
<option value=50>50
<option value=100>100
<option value=200>200
</select>

<tr>
<td bgcolor=white align=center colspan=2>
<input type="submit" value="Begin Search">
<input type="reset" value="Reset Search">

</table>

<p>

EOF

        print "$checkbox\n";

         print "$footer\n";
        &log("advanced","OK","-");
}

1;
::::::::::::::
analyze.pl
::::::::::::::
# Use:                 analyzes ntrs downloads
# Created:      07/22/2003 MLN
########################################################################

sub analyze
{
        $usage_log_root = "/usr/local/web/htdocs/ntrs/ntrs/_log.pkg/usage/days";

         &http_header("text/html");
        $date1 = $in{"date1"} if ($in{"date1"} =~ /^[\d-]+$/);
        $date2 = $in{"date2"} if ($in{"date2"} =~ /^[\d-]+$/);
        $GetRecord =
"http://ntrs.nasa.gov/?verb=GetRecord&metadataPrefix=oai_dc&identifier=";
        my($base) = &MyBaseUrl;
        $tempfile = "/tmp/ntrs-analyze.$$";
        if ($in{"host_limit"} =~ /^[\d]+$/) {
                $head_host = "| head -$in{\"host_limit\"}";
                $host_limit = $in{"host_limit"};
        } else {
                $host_limit ="";
        }
        if ($in{"id_limit"} =~ /^[\d]+$/) {
                $head_id = "| head -$in{\"id_limit\"}";
                $id_limit = $in{"id_limit"};
        } else {
                $id_limit ="";
        }
        # $id_re and $host_re are null unless something valid is passed in
        if ($in{"id_re"} =~ /^[\w\.\$\^]+$/) {
                $id_re = $in{"id_re"};
        }
        if ($in{"host_re"} =~ /^[\w\.\$\^]+$/) {
                $host_re = $in{"host_re"};
        }

         use DBI;
         $datasource = "DBI:mysql:database=ntrs;host=localhost";
         $user = "mln";
         $password = "ntrsr0cks";
         $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
                 || db_error("$DBI::errstr");

        # figure out which days to process

        opendir(D,"$usage_log_root");
        @files = sort(grep(!/^\.\.?/,readdir(D)));
        closedir(D);

        if ( ($date1) && ($date2) ) {
                foreach $d (@files) {
                        if ( ($d le $date2) && ($d ge $date1) ) {
                                push(@days,$d);
                        }
                }
        } elsif ($date1) {
                foreach $d (@files) {
                        if ($d =~ /$date1/) {
                                push(@days,$d);
                        }
                }
        } else {
                $initial_page = 1;
        }

        # figure our what days are needed
        chdir("$usage_log_root");
        open(T,">>$tempfile");
        foreach $day (@days) {
                open (D,"$day");
                while (<D>) {
                        # the format for these files is:
                        # host yyyy-mm-dd oaiID
                        $line = $_;
                        chomp($line);
                        ($host,undef,$id) = split(/ /,$line);
                        if ( ($host_re) && ($id_re) ) {
                                # test both re's
                                if ( ($host =~ /$host_re/i) &&
                                     ($id =~ /$id_re/i) ) {
                                        print T $_;
                                }
                                next;
                        }
                        if ($host_re) {
                                # test just host_re
                                if ($host =~ /$host_re/i) {
                                        print T $_;
                                }
                                next;
                        }
                        if ($id_re) {
                                # test just id_re
                                if ($id =~ /$id_re/i) {
                                        print T $_;
                                }
                                next;
                        }
                        # no conditions if we're here; write the original line
                        print T $_;
                }
                close(D);
        }
        close(T);

        $top_ids = `cat $tempfile | awk \'{print \$3}\' | sort | uniq -c | sort -r
$head_id`;
```

```
        $top_hosts = `cat $tempfile | awk \'{print \$1}\' | sort | uniq -c | sort -r
$head_host`;

        $total_downloads = `wc -l $tempfile`;
        $total_downloads =~ s#/.*$##;

        $total_unique_hosts = `cat $tempfile | awk \'{print \$1}\' | sort -u | wc -l`;
        $total_unique_hosts =~ s#/.*$##;

        $total_unique_ids = `cat $tempfile | awk \'{print \$3}\' | sort -u | wc -l`;
        $total_unique_ids =~ s#/.*$##;

        $total_days = scalar(@days);

        if ( ($total_downloads > $total_days) && ($total_days > 0) ) {
                $downloads_per_day = int($total_downloads / $total_days);
        } elsif ($total_days == 0) {
                $downloads_per_day = "(n/a)";
        } elsif ( ($total_downloads < $total_days) && ($total_downloads > 0) ) {
                $downloads_per_day = "< 1";
        } elsif ($total_downloads == 0) {
                $downloads_per_day = "0";
        } else {
                $downloads_per_day = "(error)";
        }

        chdir("$real_current_dir");
         # get headers and footers

         open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
         while (<H>) {
                # turn on the "admin search" button
                if (/<!--analyze-->/) {
                        s/dedede/white/g;
                }
                 $header .= $_;
         }
         close(H);

         print "$header\n";

        if ($initial_page) {

                print "<form action=\"\" method=\"GET\">\n";
                print "<input type=hidden name=method value=analyze>\n";
                print "<table>\n";
                print "<tr><td>Start Date: <td><input type=text name=date1>\n";
                print "<tr><td>End Date: <td><input type=text name=date2>\n";
                print "<tr><td>Host Pattern: <td><input type=text name=host_re>\n";
                print "<tr><td>Host Limit: <td><input type=text name=host_limit
value=20>\n";
                print "<tr><td>ID Pattern: <td><input type=text name=id_re>\n";
                print "<tr><td>ID Limit: <td><input type=text name=id_limit value=20>\n";
                print "</table><p>\n";
                print "<input type=submit value=\"Analyze log files\">";
                print "<input type=reset value=\"Reset analysis\">";
                print "</form>\n";
                print "<p><i>Dates are of the form:  YYYY-MM-DD or YYYY-MM or YYYY<br>
Patterns are Perl regular expressions</i>\n";

        } else {
                $date_range = "$date1 ";
                $date_range .= " - $date2" if ($date2);

                print "<table cellpadding=5 cellspacing=5>\n";
                print "<th colspan=2 bgcolor=dedede>NTRS Usage Summary for $date_range";
                print "<br>host pattern = $host_re" if ($host_re);
                print "<br>id pattern = $id_re" if ($id_re);
                print "</th>\n";
                print "<tr><td>Total downloads: <td>$total_downloads\n";
                print "<tr><td>Unique downloads: <td>$total_unique_ids\n";
                print "<tr><td>Total days: <td>$total_days\n";
                print "<tr><td>Downloads per day: <td>$downloads_per_day\n" if
($total_days > 1);
                print "<tr><td>Unique hosts: <td>$total_unique_hosts\n";
                if ($id_limit ne "") {
                        print "<tr><td><a href=\"#top_ids\">Top $id_limit documents</a>";
                } else {
                        print "<tr><td><a href=\"#top_ids\">All documents</a>";
                }
                if ($host_limit ne "") {
```

```
			print "<td><a href=\"#top_hosts\">Top $host_limit hosts</a>";
		} else {
			print "<td><a href=\"#top_hosts\">All hosts</a> ";
		}
		print "</table>\n";

		print "<p><a name=top_ids>\n";

		print "<table border=1 width=70%>\n";
		if ($id_limit ne "") {
			print "<th colspan=2 bgcolor=dedede>Top $id_limit Documents ";
		} else {
			print "<th colspan=2 bgcolor=dedede>All Documents ";
		}
		print "<br>(pattern = $id_re)" if ($id_re);
		print "</th>\n";
		print "<tr><th>downloads<th>documents\n";
		# the line below should only modify the first column
		$top_ids =~ s/\d+\s+oai:/<tr><td>$&/g;
		#$top_ids =~ s#oai.*#<td><a href=$GetRecord$&>$&</a>#g;
		foreach $i (split/\n/,$top_ids) {
			chomp $i;
			$id = $i;
			$id =~ s#\s*<tr><td>\d+\s+##;

			$sql = "SELECT title, type FROM metadata WHERE oaiID=\"$id\"";
			 $sth = $dbh->prepare($sql);
			$sth->execute;
			($title,$type) = $sth->fetchrow_array;
			$sth->finish;
			if ($id =~ /\+/) {
				# if the oaiID has a "+" in it, it screws
				# up the substitution
				$tempid = $id;
				$urlid = $id;
				$tempid =~ s/\+/\\\+/g;
				$urlid =~ s/\+/%2B/g;
				$top_ids =~ s#$tempid#<td>$title, $type (<a
href=$GetRecord$urlid>$id</a>)#g;
			} else {
				$top_ids =~ s#$id#<td>$title, $type (<a
href=$GetRecord$id>$id</a>)#g;
			}
		}
		print "$top_ids\n";
		print "</table>\n";

		print "<p><a name=top_hosts>\n";

		print "<table border=1>\n";
		if ($host_limit ne "") {
			print "<th colspan=2 bgcolor=dedede>Top $host_limit Hosts ";
		} else {
			print "<th colspan=2 bgcolor=dedede>All Hosts ";
		}
		print "<br>(pattern = $host_re)" if ($host_re);
		print "</th>\n";
		print "<tr><th>downloads<th>IP name / addr\n";
		$top_hosts =~ s/\d+ /<tr><td>$&<td>/g;
		print "$top_hosts\n";
		print "</table>\n";
		unlink($tempfile);
	}

	open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
	while (<F>) {
		$footer .= $_;
	}
	close(F);

	print "$footer\n";
	&log("analyze","OK","-");
}

1;
::::::::::::::
browse.pl
::::::::::::::
########################################################################
# Method:	browse
# Use:		browse page for NTRS
# Created:      09/02/2002 MLN
# Updated:     06/19/2003 MLN - uses the static archives html page
#########################################################################

sub browse
{
        &http_header("text/html");
       my($base) = &MyBaseUrl;

        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
               # turn on the "browse search" button
               if (/<!--browse-->/) {
                       s/dedede/white/g;
               }
                $header .= $_;
        }
        close(H);

        open(H,"ntrs.pkg/radio.html") || die ("$!: cannot open radio file\n");
        while (<H>) {
                $radio .= $_;
        }
        close(H);


        open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
        while (<F>) {
                $footer .= $_;
        }
        close(F);

        print "$header\n";
        print <<EOF;


<form action=$base method=GET>
<input type=hidden name=method value=search>
<input type=hidden name=offset value=0>
<input type=hidden name=mode value=browse>
<input type=hidden name=boolean value=or>
<input type=hidden name=boolean_terms value=or>
EOF
        print "$radio\n";

       print <<EOF;
<p>

<table>
<td align=right width=30%> <label for="order by">Order results by:</label>
<td> <select name=orderby id="order by">
<option selected value=date>Publication date
<option value=dateStamp>Date added to NTRS
<option value=title>Title
<option value=creator>Author(s)
<option value=type>Citation
</select>

<tr>
<td align=right width=30%> <label for="order results">Order of results:</label>
<td> <select name=order id="order results">
<option selected value=DESC>Descending
<option value=ASC>Ascending
</select>

<tr>
<td align=right width=30%> <label for="results per page">Results per page:</label>
<td> <select name=limit id="results per page">
<option value=10>10
<option selected value=25>25
<option value=50>50
<option value=100>100
<option value=200>200
</select>

<tr>
<td> &nbsp;

<tr>
```

```
<td bgcolor=white align=center colspan=2>
<input type="submit" value="Begin Browse">
<input type="reset" value="Reset Browse"><p>
<p>
<!--
<input type=checkbox name=debug value=on>Debug info
-->
</form>
</table>
EOF

        print "$footer\n";
        &log("browse","OK","-");
}

1;
::::::::::::::
feedback.pl
::::::::::::::
########################################################################
# Method:       feedback
# Use:                  feedback page for NTRS
# Created:      10/10/2002 MLN
# Notes:        based on the NIX survey code from Bill von Ofenheim
#               w.h.c.vonofenheim\@larc.nasa.gov
########################################################################

sub feedback
{
        &http_header("text/html");
        $base = &MyBaseUrl;

        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
                # turn on the "feedback search" button
                if (/<!--feedback-->/) {
                        s/dedede/white/g;
                }
                $header .= $_;
        }
        close(H);

        print "$header\n";

        open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
        while (<F>) {
                $footer .= $_;
        }
        close(F);

        require "$real_current_dir/ntrs.pkg/survey-support.pl";

        if($ENV{CONTENT_LENGTH} == 0) {
                &PrintForm;
        }else{
                #&ReadParse;
                foreach $i (0 .. $#Statements) {
                        $name=sprintf("Q%02d",$i);
                        push(@answers,$in{$name});
                }
                &WriteResult(@answers);
                &PrintSuccessMsg;
        }


        print "$footer\n";
        &log("feedback","OK","-");
}


##############################################################################

sub PrintSuccessMsg {
print <<EOF;

<CENTER>
You have successfully submitted the feedback.  Thank you for your participation.<P>
<P>
Please select a function above to continue using NTRS.
```

```
<p>
</CENTER>
EOF
}

##############################################################################

sub PrintForm {
  local($i,$n,$name,@color,$separator);
  @color=("#FFFFFF","#FF0000","#FF9900",
          "#FFFF00","#99FF00","#00FF00");

  print <<EOF;

<CENTER>
<H2>NTRS Feedback Form</H2>

<blockquote>
Please provide feeback on using the NTRS. Federal government personnel are
requested to provide feedback; non-federal government personnel are
welcome to respond but are not required to provide comments. Please read
each statement below and rate
each according to the scale by mouse clicking in the appropriate checkbox.
Hit the <I>Submit Feedback</I> button when you are finished.
If you would like to expand on your responses to the feedback form,
please email <A
HREF="mailto:ntrs+feedback\@larc.nasa.gov">ntrs+feedback\@larc.nasa.gov</A>.
</blockquote>
</FONT>
<FORM ACTION="" METHOD="POST">
<input type=hidden name="method" value="feedback">
<TABLE BORDER=0 CELLPADDING=5 CELLSPACING=0 BGCOLOR="#FFFFFF"><TR><TD>
EOF
  foreach $i (0 .. $#Statements) {
    if($i==6) {
      print <<EOF;
(If you used the NTRS comment form to submit a question<BR>or comment then please answer
the following questions.)<p>
EOF
    }
    $name=sprintf("Q%02d",$i);
    print <<EOF;
<fieldset>
<B><legend>$Statements[$i] :</legend></B>
<dl>
<dd><label for="No Opinion"><INPUT TYPE=radio NAME=$name VALUE="0" CHECKED id="No
Opinion">No Opinion</label>
<dd><label for="Very Unsatisfied"><INPUT TYPE=radio NAME="$name" VALUE="1" id="Very
Unsatisfied">Very Unsatisfied</label>
<dd><label for="Unsatisfied"><INPUT TYPE=radio NAME="$name" VALUE="2"
id="Unsatisfied">Unsatisfied</label>
<dd><label for="Satisfied"><INPUT TYPE=radio NAME="$name" VALUE="3"
id="Satisfied">Satisfied</label>
<dd><label for="Very Satisfied"><INPUT TYPE=radio NAME="$name" VALUE="4" id="Very
Satisfied">Very Satisfied</label>
<dd><label for="Extremely Satisfied"><INPUT TYPE=radio NAME="$name" VALUE="5"
id="Extremely Satisfied">Extremely Satisfied</label>
</dl>
</fieldset>
EOF
  }
  print <<EOF;
</TD></TR></TABLE>
<INPUT TYPE="submit" VALUE="Submit Feedback">
<INPUT TYPE="reset" VALUE="Reset"><p>
</FORM>
<!--
<i><font size="-1" color="#ffffff">
If you are a repeat customer and prefer not to complete a
feedback form each time you use our services, we will assume <br>
that your request has been satisfied unless you notify us
that the service was excellent or needs improvement.</font></i>
<hr>
-->
EOF
}

1;
::::::::::::::
growth.pl
::::::::::::::
```

```
# Use:                  charts the growth of ntrs archives
# Created:      08/07/2003 MLN
########################################################################

sub growth
{
         &http_header("text/html");
        $date1 = $in{"date1"} if ($in{"date1"} =~ /^[\d-]+$/);
        $date2 = $in{"date2"} if ($in{"date2"} =~ /^[\d-]+$/);
        $archives = $in{"archives"};
        $debug = $in{"debug"};
        my($base) = &MyBaseUrl;
        my($full) = &MyFullUrl;
        $grey = "dedede";

        if ( $full =~ /archives=/ ) {
                undef($initial_page);
        } else {
                $initial_page = 1;
        }

         use DBI;
         $datasource = "DBI:mysql:database=ntrs;host=localhost";
         $user = "mln";
         $password = "ntrsr0cks";
         $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
                 || db_error("$DBI::errstr");

         # get headers, footers & select html

         open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
         while (<H>) {
                # turn on the "admin search" button
                if (/<!--analyze-->/) {
                        s/dedede/white/g;
                }
                 $header .= $_;
         }
         close(H);

         open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
         while (<F>) {
                 $footer .= $_;
         }
         close(F);

        open(N,"ntrs.pkg/select.html") || die ("$!: cannot open select\n");
        while (<N>) {
                $select .= $_;
        }
        close(N);

         print "$header\n";

        if ($initial_page) {

                print "<form action=\"\" method=\"GET\">\n";
                print "<input type=hidden name=method value=growth>\n";
                print "<table>\n";
                print "<tr><td>Start Date: <td><input type=text name=date1>\n";
                print "<tr><td>End Date: <td><input type=text name=date2>\n";
                print "<tr><td>Archives: <td>$select\n";
                print "</table><p>\n";
                print "<input type=submit value=\"Analyze archive growth\">";
                print "<input type=reset value=\"Reset analysis\">";
                print "</form>\n";
                print "<p><i>Dates are of the form:  YYYY-MM-DD or YYYY-MM or YYYY</i>\n";

        } else {
                # normalize date types of YYYY and YYYY-MM to be YYYY-MM-DD
                if ($date1 =~ /^\d\d\d\d-\d\d$/) {
                        $date1 .= "-01";
                } elsif ($date1 =~ /^\d\d\d\d$/) {
                        $date1 .= "-01-01";
                }
                if ($date2 =~ /^\d\d\d\d-\d\d$/) {
                        $date2 .= "-01";
                } elsif ($date2 =~ /^\d\d\d\d$/) {
                        $date2 .= "-01-01";
                }
                print "<h2>Changes in Repository Holdings </h2><p>\n";
                $sql = " SELECT day, size, longName FROM harvests, archives \
                        WHERE ( (harvests.archiveID = archives.archiveID) ";
                $sql .= " AND (day >= \"$date1\") " if $date1;
                $sql .= " AND (day =< \"$date2\") " if $date2;
                $sql .= " AND (shortName = \"$archives\") " if $archives;
                $sql .= " ) ORDER BY longName, day ASC  ";

                print "<font color=red>sql = $sql</font> <br>\n" if ($debug);
                $sth = $dbh->prepare($sql);
                $sth->execute;

                $no_results = 1;
                while ( ($day,$size,$longName) = $sth->fetchrow_array) {
                        undef($no_results);
                        # handle change of repository
                        if ($longName ne $oldLongName) {
                                print "</table><p>\n" if $oldLongName;
                                print "<table width=400><th colspan=3 bgcolor=$grey><font
size=+1>$longName</font></th><tr>\n";
                                print "<th>Date</th><th>Size</th><th>Change</th>\n";
                                undef($oldDay);
                                undef($oldSize);
                                undef($oldLongName);
                        }

                        # figure out the change, initial values get "-"
                        if ($oldSize) {
                                $change = $size - $oldSize;
                        } else {
                                $change = "-";
                        }
                        print "<tr align=center><td>$day<td>$size<td>$change\n";

                        # save the previous values
                        ($oldDay,$oldSize,$oldLongName) =
                                ($day,$size,$longName);

                }

                print "</table>\n";
                $sth->finish;
                if ($no_results) {
                        print "no harvest data is available ";
                        print "from $date1 " if ($date1);
                        print "to $date2 " if ($date2);
                        if ($archives eq "") {
                                print "for all archives";
                        } else {
                                print "for $archives";
                        }
                }
        }

        print "$footer\n";
        &log("growth","OK","-");
}

1;
::::::::::::::
help.pl
::::::::::::::
##########################################################################
# Method:       help
# Use:                  help page for NTRS
# Created:      05/07/2002 MLN
##########################################################################

sub help
{
        &http_header("text/html");
        my($base) = &MyBaseUrl;
        $grey="dedede";

        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
                # turn on the "help search" button
                if (/<!--help-->/) {
                        s/$grey/white/g;
                }
                 $header .= $_;
        }
        close(H);

        print "$header\n";

        open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
        while (<F>) {
                $footer .= $_;
        }
        close(F);

        if ($ENV{"HTTP_REFERER"} =~ /advanced/) {
               $header = "Advanced Search Help";
        } elsif ($ENV{"HTTP_REFERER"} =~ /browse/) {
               $header = "Browse Help";
        } elsif ($ENV{"HTTP_REFERER"} =~ /simple/) {
               $header = "Simple Search Help";
        } else {
               $header = "Search Help";
        }

       print <<EOF;

<h2>$header</h2>
<p>
<font size=-1>
(Questions? Contact the <A href="http://www.sti.nasa.gov/help/help.html">NASA STI Help
Desk</a>)
</font>
</center>
<blockquote>
EOF

       if ($ENV{"HTTP_REFERER"} =~ /advanced/) {
               print $advanced;
               print $simple;
               print $browse;
       } elsif ($ENV{"HTTP_REFERER"} =~ /browse/) {
               print $browse;
               print $simple;
               print $advanced;
       } else {
               print $simple;
               print $advanced;
               print $browse;
       }

       print <<EOF;
</blockquote>
<center>

EOF

        print "$footer\n";
        &log("help","OK","-");

}

############################################################################
# text for browse help
############################################################################

$browse = <<EOF;
<table width=100%>
<tr>
<td bgcolor=dedede>
<b>
Browse
</b>
</table>

<p>

A Browse search retrieves all the records of a specific collection in
NTRS.  Only one collection is browsed at a time. To learn more about
the different databases making up NTRS, go to
<a href="?method=about">About the NTRS collections</a>.

<p>
```

```
EOF

###########################################################################
# text for advanced help
###########################################################################

$advanced = <<EOF;
<table width=100%>
<tr>
<td bgcolor=dedede>
<b>Advance Search</b>
</table>

<p>

Advanced searching differs from <a href="?method=simple">Simple Search</a> because there
is no keyword searching. Users input search terms in one or more search fields to find
information. Users do not need to use an "AND" between terms because the search
automatically searches for ALL the terms in a search. Users can also choose to
narrow/broaden their searches by selecting and de-selecting the information sources they
are searching on. (see <b>Searching for NASA and non-NASA information</b> below)
<p>

Users create more accurate searches by searching on specific search fields:
<ul>
<li> Title
<li> Author(s)
<li> Dates
<li> Report number/Journal/Conference
<li> Abstract (Paragraph summarizing main points in report)
</ul>
<p>
Advanced Search <b>User Options</b>:
<ul>
<li> Change the default field search setting from "AND" to "OR" using the <b>Combine
fields</b> selection. An "AND" search finds <underline>all</underline> the words in a
search and an "OR" searches for <underline>any</underline> of the search terms. "AND"
narrows a search and "OR" broadens a search.
<li> Change the <b>Order results by</b> publication date, date added to NTRS, title,
author(s), or citation by using the drop-down menu box.
<li> Change the <b>Order of results</b> from descending (current reports listed first) to
ascending (oldest reports listed first) by using the drop-down menu box.
</ul>
<p>


<b>Search Operators</b>


<p>

There are several operators to assist in expressing your query.  These
operators can be used in either the simple or advanced search interfaces.

<ul>
<li>  A "-" indicates that the word must not be in the search results.
For example, searching on <A
href="http://ntrs.nasa.gov/index.cgi?method=search&offset=0&mode=advanced&title=&creator=
-
michael+nelson&date=&type=&description=&boolean=and&orderby=date&order=DESC&limit=25&arch
ives=atrs&archives=casi&archives=genesis.jpl.nasa.gov&archives=www.giss.nasa.gov&archives
=gtrs&archives=jtrs&archives=ktrs&archives=ltrs.larc.nasa.gov&archives=mtrs&archives=naca
.larc.nasa.gov&archives=riacs&archives=ssctrs">-michael
nelson</a> in the <b>Author field</b> finds all records with the word "nelson", but not
"michael".  This gives different results than the search <A
href="http://ntrs.nasa.gov/index.cgi?method=search&offset=0&mode=advanced&title=&creator=
michael+nelson&date=&type=&description=&boolean=and&orderby=date&order=DESC&limit=25&arch
ives=atrs&archives=casi&archives=genesis.jpl.nasa.gov&archives=www.giss.nasa.gov&archives
=gtrs&archives=jtrs&archives=ktrs&archives=ltrs.larc.nasa.gov&archives=mtrs&archives=naca
.larc.nasa.gov&archives=riacs&archives=ssctrs">michael
nelson</a>

<li> A "*" at the end of the word is used for stemming.  For example, searching on
<a
href="http://ntrs.nasa.gov/index.cgi?method=search&offset=0&mode=advanced&title=comput*&c
reator=&date=&type=&description=&boolean=and&orderby=date&order=DESC&limit=25&archives=at
rs&archives=casi&archives=genesis.jpl.nasa.gov&archives=www.giss.nasa.gov&archives=gtrs&a
rchives=jtrs&archives=ktrs&archives=ltrs.larc.nasa.gov&archives=mtrs&archives=naca.larc.n
asa.gov&archives=riacs&archives=ssctrs">comput*</a> in the <b>Title field</b>
finds all records with "computer", "computing", "computation", etc., in report titles
```

```
<li> Enclosing the terms in quotes (") specifies searching for an exact phrase.
For example, searching on <a
href="http://ntrs.nasa.gov/index.cgi?method=search&offset=0&mode=advanced&title=&creator=
&date=&type=&description=%22parallel+virtual+machine%22+&boolean=and&orderby=date&order=D
ESC&limit=25&archives=atrs&archives=casi&archives=genesis.jpl.nasa.gov&archives=www.giss.
nasa.gov&archives=gtrs&archives=jtrs&archives=ktrs&archives=ltrs.larc.nasa.gov&archives=m
trs&archives=naca.larc.nasa.gov&archives=riacs&archives=ssctrs">parallel virtual
machine</a> in the <b>Abstract field</b> finds this exact phrase

</ul>



<b>Searching for NASA and non-NASA information</b>

<p>

NTRS provides access to NASA information but it also collects scientific and technical
information from organizations, institutions, universities, and other federal agencies
with similar content. Users can select and de-select <b>NASA Archives </b> and <b>Non-
NASA Archives</b> using the check boxes beside each site. By default only the NASA
Archives are selected but users can select and de-select any of the NASA and non-NASA
sites.

<p>
<center>
<table>

<tr>
<td colspan=2 align=center>Select the databases you would like to search

<tr>
<td align=center bgcolor=dedede> NASA Archives
<td align=center bgcolor=dedede> Non-NASA Archives

<tr>
<td> <input type="checkbox" name="archives" value="atrs" CHECKED id=atrs> <label
for=atrs>Ames Research Center</label>
<td> <input type="checkbox" name="archives" value=arc id=ARC-UK> <label for=ARC-
UK>Aeronautical Research Committee (UK)</label>
<tr>
<td> <input type="checkbox" name="archives" value="casi" CHECKED id=casi> <label
for=casi>Center for AeroSpace Information (CASI)</label>
<td> <input type="checkbox" name="archives" value=arxiv.org id=arxiv.org> <label
for=arXiv.org>arXiv Physics Eprints Server</label>
<tr>
<td> <input type="checkbox" name="archives" value=genesis.jpl.nasa.gov CHECKED
id=genesis.jpl.nasa.gov> <label for=genesis.jpl.nasa.gov>GENESIS (JPL)</label>
<td> <input type="checkbox" name="archives" value=biomedcentral.com CHECKED
id=biomedcentral.com> <label for=biomedcentral.com>BioMed Central</label>
</td></tr></table></center>
<br>
<p>

<b>Advanced Search Examples</b>

<p>

1) Title and Author search:
<p>
<center>
<form action=http://ntrs.nasa.gov/index.cgi method=GET>
<input type=hidden name=method value=search>
<input type=hidden name=limit value=25>
<input type=hidden name=offset value=0>
<input type=hidden name=mode value=advanced>
<input type=hidden name=archives value=ltrs.larc.nasa.gov>
<input type=hidden name=archives value=casi>
<input type=hidden name=archives value=atrs>
<input type=hidden name=archives value=gtrs>
<input type=hidden name=archives value=jtrs>
<input type=hidden name=archives value=ktrs>
<input type=hidden name=archives value=mtrs>
<input type=hidden name=archives value=riacs>
<input type=hidden name=archives value=ssctrss>
<input type=hidden name=archives value=genesis.jpl.nasa.gov>
<input type=hidden name=archives value=www.giss.nasa.gov>
<input type=hidden name=archives value=naca.larc.nasa.gov>
<table>
<tr>
```

```
<td align=right width=30%> <label for=title>Title: </label>
<td><input name=title size=40 id="title" value="global positioning system">
<tr>
<td align=right width=30%> <label for=creator>Author(s): </label>
<td><input name=creator size=40 id="creator" value="Semion Kizhner">
</td>
<td>
</td>
</tr>
<tr><td></td><td><input type=submit id="see search results" value="See search
results"></td></tr>
</table>
</form>
</center>


<ul>
<i><b>Note:</b> Author names can be input first name last name or last name first name.
No punctuation is necessary between first and last names.</i>
</ul>

<p>

2) Author and Date search:
<p>
<center>
<form action=http://ntrs.nasa.gov/index.cgi method=GET>
<input type=hidden name=method value=search>
<input type=hidden name=limit value=25>
<input type=hidden name=offset value=0>
<input type=hidden name=archives value=ltrs.larc.nasa.gov>
<input type=hidden name=archives value=casi>
<input type=hidden name=archives value=atrs>
<input type=hidden name=archives value=gtrs>
<input type=hidden name=archives value=jtrs>
<input type=hidden name=archives value=ktrs>
<input type=hidden name=archives value=mtrs>
<input type=hidden name=archives value=riacs>
<input type=hidden name=archives value=ssctrss>
<input type=hidden name=archives value=genesis.jpl.nasa.gov>
<input type=hidden name=archives value=www.giss.nasa.gov>
<input type=hidden name=archives value=naca.larc.nasa.gov>
<input type=hidden name=mode value=advanced>
<table>
<tr>
<td align=right width=30%> <label for=title>Title: </label>
<td><input name=title size=40 id="title">
<tr>
<td align=right width=30%> <label for=creator>Author(s): </label>
<td><input name=creator size=40 value="lawrence green" id="creator">
<tr>
<td align=right width=30%> <label for=date>Date: </label>
<td><input name=date size=40 value="2001" id="date">
</td>
</tr>
<tr><td></td><td><input type=submit id="see search results" value="See search
results"></td></tr>
</table>
</form>
</center>
<p>
<ul>
<i><b>Note:</b> Search "date" using a single year only as was done in this example.</i>
</ul>
<p>

3) Report Number/Journal/Conference search:
<p>

<center>
<form action=http://ntrs.nasa.gov/index.cgi method=GET>
<input type=hidden name=method value=search>
<input type=hidden name=limit value=25>
<input type=hidden name=archives value=ltrs.larc.nasa.gov>
<input type=hidden name=archives value=casi>
<input type=hidden name=archives value=atrs>
<input type=hidden name=archives value=gtrs>
<input type=hidden name=archives value=jtrs>
<input type=hidden name=archives value=ktrs>
<input type=hidden name=archives value=mtrs>
<input type=hidden name=archives value=riacs>
```

```
<input type=hidden name=archives value=ssctrss>
<input type=hidden name=archives value=genesis.jpl.nasa.gov>
<input type=hidden name=archives value=www.giss.nasa.gov>
<input type=hidden name=archives value=naca.larc.nasa.gov>
<input type=hidden name=offset value=0>
<input type=hidden name=mode value=advanced>
<table>
<tr>
<td align=right width=30%> <label for=title>Title:</label>
<td><input name=title size=40 value="navier stokes" id="title">
<tr>
<td align=right width=30%> <label for=creator>Author(s):</label>
<td><input name=creator size=40 id="creator">
<tr>
<td align=right width=30%> <label for=date>Date:</label>
<td><input name=date size=40 id="date">
<tr>
<td align=right width=30%> <label for=type>Report Number / <br>Journal /
Conference:</label>
<td><input name=type size=40 value="aiaa*" id="type">
</td>
</tr>
<tr><td></td><td><input type=submit id="see search results" value="See search
results"></td></tr>
</table>
</form>
</center>

<ul>  <i><b>Note:</b> Report numbers should not have any punctuation
like hyphens and dashes. Use the    <b>Report Number/Journal/Conference</b>
search to find report numbers, journal titles, and  conference proceedings.</i>  </ul>




EOF

############################################################################
# text for simple help
############################################################################

$simple = <<EOF;
<table width=100%>
<tr>
<td bgcolor=dedede>
<b>Simple Search</b>
</table>

<p>

The Simple Search is a keyword search of NASA scientific and technical information.
Simple Search differs from the <a href="?method=advanced">Advanced Search</a> because it
is a general search and it searches only NASA information. Users do not need to use an
"AND" between terms because the search automatically searches for ALL the terms in a
search. Simple Search can be used to find:
<ul>
<li> Titles
<li> Authors
<li> Dates
<li> Reports
<li> Abstracts (Paragraph summarizing main points in report)
</ul>



<b>Simple Search Examples</b>


<ul>

<li> <a
href="http://ntrs.nasa.gov/index.cgi?method=search&limit=40&offset=0&mode=simple&order=DE
SC&keywords=neural+networks&boolean=and">neural networks</a> finds publications with all
search terms but not
necessarily in any particular order

<li> <a
href="http://ntrs.nasa.gov/index.cgi?method=search&limit=40&offset=0&mode=simple&order=DE
SC&keywords=madaras+nondestructive+testing&boolean=and">madaras nondestructive
testing</a> retrieves records about
nondestructive testing by author with the last name of madaras
```

```
<li> <a
href="http://ntrs.nasa.gov/index.cgi?method=search&limit=40&offset=0&mode=simple&order=DE
SC&keywords=mars+lander+2002&boolean=and">mars lander 2002</a> searches for records about
mars and lander and
the year 2002


<li><a
href="http://ntrs.nasa.gov/index.cgi?method=search&limit=25&offset=0&mode=simple&order=DE
SC&keywords=NASA+CR+147848">NASA CR 147848</a> and <a
href="http://ntrs.nasa.gov/index.cgi?method=search&limit=25&offset=0&mode=simple&order=DE
SC&keywords=AIAA+96+3991">AIAA 96 3991</a> search for the specific reports NASA-CR-147848
and AIAA Paper 96-3991. Do not use any punctuation if searching for reports.


<li> Using numbers in a search retrieves records that have the
number anywhere in the record such as in the publication year or
in the report number.

</ul>

EOF

1;
::::::::::::::
news.pl
::::::::::::::
########################################################################
# Method:       news
# Use:                  news page for NTRS
# Created:      05/07/2002 MLN
########################################################################

sub news
{
        &http_header("text/html");
        my($base) = &MyBaseUrl;

        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
                # turn on the "news search" button
                if (/<!--news-->/) {
                        s/dedede/white/g;
                }
                 $header .= $_;
        }
        close(H);

        print "$header\n";

        open(N,"ntrs.pkg/news.rss") || die ("$!: cannot open news\n");
        while (<N>) {
                $_ =~ s/&lt;/</g;
                $_ =~ s/&gt;/>/g;
                 push(@news,$_);
        }
        close(N);

        print "<table cellpadding=5>\n";
        # process the RSS file
        undef($title);
        undef($item);
        undef($description);

        foreach $line (@news) {

                next unless (($line =~ /<item>/) || ($item));
                $item=1;

                if ($line =~ /<description>/i) {
                        $description .= $line;
                        $description_num++;
                } elsif ($description) {
                        $description .= $line;
                }
                if ($line =~ /<title>/i) {
                        $title .= $line;
                        $title_num++;
```

```
		} elsif ( ($title) && !($description) ) {
			$title .= $line;
		}
		if ($line =~ /<\/title>/) {
			# print $title;
			$title =~ s/<title>//g;
			$title =~ s#</title>##g;
		}
		if ($line =~ /<\/description>/) {
			# print $title;
			$description =~ s/<description>//ig;
			$description =~ s#</description>##ig;
			#if ($title_num % 2) {
				#print "<tr><td>$title <br>
<blockquote>$description</blockquote>\n";
			#} else {
			#	print "<tr bgcolor=dedede><td>$title <br>
<blockquote>$description</blockquote>\n";
			#}
			print "<tr><td bgcolor=dedede>$title <tr><td>
<blockquote>$description</blockquote>\n";

			undef($title);
			undef($item);
			undef($description);
		}
	}
	print "</table>\n";

	open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
	while (<F>) {
		$footer .= $_;
	}
	close(F);

	print "$footer\n";
	&log("news","OK","-");
}

1;
::::::::::::::
ordering.pl
::::::::::::::
###########################################################################
# Method:	ordering
# Use:			ordering page for NTRS
# Created:	05/07/2002 MLN
###########################################################################

sub ordering
{
	&http_header("text/html");
	my($base) = &MyBaseUrl;
	my($id) = $in{"oaiID"};
	my($casi_doc_id);

	# get headers and footers

	open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
	while (<H>) {
		# turn on the "ordering" button
		if (/<!--ordering-->/) {
			s/dedede/white/g;
		}
		$header .= $_;
	}
	close(H);

	print "$header\n";

	open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
	while (<F>) {
		$footer .= $_;
	}
	close(F);

	use DBI;

	$datasource = "DBI:mysql:database=ntrs;host=localhost";
	$user = "mln";
	$password = "ntrsr0cks";
```

```
        #$dbh = DBI->connect($datasource, $user, $password, {'RaiseError' => 1});
       $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
               || db_error("$DBI::errstr");

        $sql = "SELECT longName, nasa, url, title, creator, date, identifier, \
       source, format, publisher, relation, type FROM metadata, archives \
       WHERE (oaiID=\"$id\") AND (metadata.archiveID = archives.archiveID)";

        $sth = $dbh->prepare($sql);
       $sth->execute;

       ($longName, $nasa, $url, $title, $creator, $date, $identifier,
        $source, $format, $publisher, $relation,$type) = $sth->fetchrow_array;

       if ($nasa eq "nasa" || (!$oaiID && !$nasa)) {
               (undef,undef,$casi_doc_id) = split(":",$id) if ($longName =~ /CASI/);
               print "<h2>Ordering From the Center for AeroSpace Information
(CASI)</h2>\n";
               print "<blockquote>\n";
               print "<table border=0>\n";
               print "<tr><td bgcolor=dedede>Title:<td>$title\n" if ($title);
               print "<tr><td bgcolor=dedede>Author(s):<td>$creator\n" if ($creator);
               print "<tr><td bgcolor=dedede>Date:<td>$date\n" if ($date);
               print "<tr><td bgcolor=dedede>Citation:<td>$type\n" if ($type);
               print "<tr><td bgcolor=dedede>Doc I.D.:<td>$casi_doc_id\n" if
($casi_doc_id);
               print "<tr><td bgcolor=dedede>Price Code(s):<td><a
href=\"http://www.sti.nasa.gov/price1.pdf\">$relation</a>\n" if (($relation) &&
($casi_doc_id));
               print "<tr><td bgcolor=dedede>Order Form:<td><a target=_new
href=\"https://www.sti.nasa.gov/cgi-bin/ordersti.pl\">https://www.sti.nasa.gov/cgi-
bin/ordersti.pl</a>\n";
               print "</table>\n";
               print "</blockquote>\n";

       print <<EOF;
<blockquote>
You may also contact CASI directly at:
</blockquote>
<p>
<pre>
NASA Center for AeroSpace Information (CASI)
7121 Standard Drive
Hanover, MD 21076-1320
Phone: (301) 621-0390
E-mail: help\@sti.nasa.gov
Website:  <a href="http://www.sti.nasa.gov/">www.sti.nasa.gov</a>
</pre>
EOF
       } else {

               print "<h2>This is Not a NASA Document</h2>\n";
               print "You must contact the maintainers of the originating archive for
hardcopy ordering information <p>\n";
               print "<table border=0>\n";
               print "<tr><td>Title:<td>$title\n" if ($title);
               print "<tr><td>Author(s):<td>$creator\n" if ($creator);
               print "<tr><td>Date:<td>$date\n" if ($date);
               print "<tr><td>Citation:<td>$type\n" if ($type);
               print "<tr><td>Originating Archive:<td>$longName\n" if ($longName);
               print "<tr><td>Archive URL:<td><a href=$url>$url</a>\n" if ($url);
               print "</table>\n";

       }

        $sth->finish;
        $dbh->disconnect;

        print "$footer\n";
       &log("ordering","OK","-");
}

1;
::::::::::::::
privacy.pl
::::::::::::::
####################################################################
# Method:       privacy
# Use:                  privacy page for NTRS
# Created:      05/07/2002 MLN
```

```
########################################################################

sub privacy
{
        &http_header("text/html");
        my($base) = &MyBaseUrl;

        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
               # turn on the "privacy search" button
               if (/<!--privacy-->/) {
                      s/dedede/white/g;
               }
                $header .= $_;
        }
        close(H);

        print "$header\n";

        open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
        while (<F>) {
                $footer .= $_;
        }
        close(F);

        print <<EOF;

<h2>
NASA Privacy Statement, Disclaimer, and Accessibility Certification
</h2>
<a href="http://www.hq.nasa.gov/privacy.html">http://www.hq.nasa.gov/privacy.html</a>

<h2>
Copyright
</h2>

Most the information at this site is in the public domain. Unless
otherwise stated, these documents may be freely distributed and used for
non-commercial, scientific, educational or personal purposes. However,
you may encounter documents or portions of documents contributed by
private companies or organizations. Other parties may retain all rights
to publish or reproduce these documents. Commercial use of the documents
on this site may be protected under U.S. and foreign copyright laws.

<center>
EOF

        print "$footer\n";
        &log("privacy","OK","-");
}

1;
::::::::::::::
recommendation.pl
::::::::::::::
#
# Method:      recomendation
# Use:         using Johan Bollen's Hebbian learning method, take an oaiID
#              and return a list of 10 "recommendations" for the oaiID
# created:     2003-07-31 MLN

sub recommendation {
        use LWP::Simple;
        use DBI;
        $sourceID = $in{"oaiID"};
        $dean_url =
"http://mercury.seven.research.odu.edu:7704/servlet/NTRS_spreadAct\?query=";
        $datasource = "DBI:mysql:database=ntrs;host=localhost";
        $user = "mln";
        $password = "ntrsr0cks";
        $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
|| db_error("$DBI::errstr");
        $grey = "dedede";
        $white = "white";

        &http_header("text/html");

        $result = get("$dean_url$sourceID");
        @temp = split(/\n/,$result);

        foreach $id (@temp) {
                next unless ($id =~ /^oai.*/);
                $id =~ s#<br></n>##g;
                push(@recommended,$id);
        }

        $sql = "SELECT title, type FROM metadata archives WHERE oaiID=\'$sourceID\'";
        $sth = $dbh->prepare($sql);
         $sth->execute;
        ($title, $type) = $sth->fetchrow_array;

        print "<center>Recommendations For Documents Related To: <br><h2>$title,
$type</h2><p>\n";

        $count = 1;
        print "<table cellpadding=5 cellspacing=0 border=0 width=95%>";
        foreach $id (@recommended) {
                 # change color every other row
                 if ( ($count % 2) == 0 ) {
                         $color = $white;
                 } else {
                         $color = $grey;
                 }
                $sql = "SELECT title, creator, type, date, identifier, dateStamp,
longName, description FROM metadata, archives WHERE (oaiID=\'$id\' AND
(metadata.archiveID = archives.archiveID))";
                $sth = $dbh->prepare($sql);
                $sth->execute;
                while (
($title,$creator,$type,$date,$identifier,$datestamp,$longName,$description) = $sth-
>fetchrow_array )
                {
                        # identifiers could have been concatenated
                        # split, hyperlink, rejoin
                        # note the extra space on the split
                        @identifiers = split("; ",$identifier);

                        foreach (@identifiers) {
                                # only hyperlink http:// or ftp:// identifiers
                                next unless (/(ht|f)tp:\/\//);
                                # some DOIs have "<" and ">" chars in them!
                                s/\s*//g;
                                # semi-colons in the URL are not "split" on and
                                # must be encoded (ref: some GISS DOIs)
                                s/;/%3B/g;
                                s/.*/<a
href=\"?method=display&redirect=$&&oaiID=$oaiID\">$&<\/a>/;
                        }
                        $identifier = join(" <br>\n ",@identifiers);

                        # longName doesn't get set in browse mode...
                        $longName = $browse_longname if (!$longName) ;

                        # remove this after the OK from JoAnne
                        if ( ($longName =~ /^NASA Ames/) ||
                        ($longName =~ /^NASA Stennis/) ||
                        ($longName =~ /^NASA Goddard Space/) ||
                        ($longName =~ /^NASA Kennedy/) ) {
                                undef($identifier);
                        }

                        if (!$identifier) {
                                $identifier = "<a href=?method=ordering&oaiID=$oaiID>No
Digital Version Available - Order This Document</a>";
                        }

                        print <<EOF;
<tr><td bgcolor=$color valign=top>$count.</td>
<td bgcolor=$color>$title<font size=-1><i><br>$creator<br>$longName<br>$type
EOF
                         if ($type =~ /$date/) {
                                # date is part of type, so print nothing
                        } elsif (($type) && ($date)) {
                                print ", $date";
                        } elsif ($date) {
                                print "$date";
                        }
                        print <<EOF;
</i>
<br>$description
<br>$identifier
<br>Updated/Added to NTRS: $datestamp
</font></td>
EOF
                        $count++;
                }
                $sth->finish;
        }

        # populate a no results mesg if we had no results; make it blank
        # if we did have results (so the print<<EOF doesn't change)
        if (@recommended) {
                $no_results ="";
        } else {
                $no_results = "No recommendations are currently available ";
                $no_results .= "for this document.  This occurs when the ";
                $no_results .= "document has not been viewed, or has been ";
                $no_results .= "added in the last month.  Recommendations ";
                $no_results .= "are recomputed early each month, so please ";
                $no_results .= "check back for recommendations about this ";
                $no_results .= "document.";
        }

print <<EOF;
</table>
<p>
<blockquote>$no_results</blockquote>
<p>
<img src="http://ntrs.nasa.gov/images/z.gif" alt="Zeitgeist Logo">
<p>
<a target=_blank href="http://www.cs.odu.edu/~jbollen/dean/">Recommendations
by Zeitgeist</a>
<p>
<font size=-1>
<a href="javascript:window.close();">Close This Window</a>
</font>
<p>
</center>
EOF
        # only disconnect if we actually got results from the database
        $dbh->disconnect if (@recommended);

        &log("recommendation","OK","oaiID=$sourceID");

}

1;

::::::::::::::
results.pl
::::::::::::::
#######################################################################
# Method:        results
# Use:                  results page for NTRS
# Created:      10/10/2002 MLN
# Notes:        based on the NIX survey code from Bill von Ofenheim
#               w.h.c.vonofenheim\@larc.nasa.gov
#######################################################################

sub results
{
        &http_header("text/html");
        $base = &MyBaseUrl;

        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
                # turn on the "results search" button
                if (/<!--results-->/) {
                        s/dedede/white/g;
                }
                 $header .= $_;
        }
        close(H);

        print "$header\n";

        require "$real_current_dir/ntrs.pkg/survey-support.pl";
        @colors=("#FFFFFF","#FF0000","#FF9900","#FFFF00","#99FF00","#00FF00");

        opendir(DIR,"$real_current_dir/ntrs.pkg/survey");
        @files=grep(/^\d\d\d\d\d\d$/,readdir(DIR));
        closedir(DIR);

        foreach $file (@files) {
                &ReadResults(1,$file);
                $rcount{$file}=$RCount;
                foreach $i (0..$#Statements) {
                        $counts{$file}[$i]=$Counts[$i];
                        $results{$file}[$i]=$Results[$i];
                }
        }

        chop($date=`date +"%m/%d/%Y %H:%M"`);

print <<EOF;

<H1>$SurveyTitle</H1>
<HR>
$date<HR>
<H3>Survey Statements</H3>
<TABLE BORDER>
EOF

        foreach $n (0 .. $#Statements) {
                $total[$n]=0;
                $qtrtotal[$n]=0;
                $yrtotal[$n]=0;
                $label=sprintf("S%02d",$n+1);
                print "<TR><TH NOWRAP>$label</TH><TD>$Statements[$n]</TD></TR>\n";
        }

print <<EOF;
</TABLE>
<P>
<H3>Survey Results</H3>
<TABLE BORDER=0 CELLSPACING=1 CELLPADDING=1><TR>
<TH BGCOLOR="#FFFFFF"> 0 = No Opinion <TH>
<TH BGCOLOR="$colors[1]"> 1 = Strongly Disagree <TH>
<TH BGCOLOR="$colors[2]"> 2 = Disagree <TH></TR><TR>
<TH BGCOLOR="$colors[3]"> 3 = Neither Disagree nor Agree <TH>
<TH BGCOLOR="$colors[4]"> 4 = Agree <TH>
<TH BGCOLOR="$colors[5]"> 5 = Strongly Agree <TH>
</TR></TABLE>
<P>
<TABLE BORDER>
<TR><TH>Date</TH><TH>Number of<BR>Responses</TH>
EOF

        foreach $n (0 .. $#Statements) {
                $label=sprintf("S%02d",$n+1);
                print "<TH NOWRAP>$label</TH>";
        }

        $rctotal=0;
        $first=1;

        print "<TH>&nbsp;</TH><TH>Avg.</TH></TR>\n";
        foreach $key (sort keys %counts) {
                ($year,$month)=$key=~/^(\d\d\d\d)(\d\d)$/;
                $month=~s/^0//;
                if(($month>=10) && ($month<=12)) {
                        $newquarter=1;
                }elsif(($month>=1) && ($month<=3)) {
                        $newquarter=2;
                }elsif(($month>=4) && ($month<=6)) {
                        $newquarter=3;
                }elsif(($month>=7) && ($month<=9)) {
                        $newquarter=4;
        }
        if($first==1) {
                $oldquarter=$newquarter;
                $rcqtrtotal=0;
                $rcyrtotal=0;
                $monthcnt=0;
                $yrmonthcnt=0;
                $first=0;
        }
        if($oldquarter != $newquarter) {
```

```
		$grandqtrtotal=0;
		print "<TR><TH ALIGN=right>Quarter : </TH><TH
ALIGN=center>$rcqtrtotal</TH>";
		foreach $n (0 .. $#Statements) {
			if($monthcnt==0) {
				$average=0;
			}else{
				$average = $qtrtotal[$n] / $monthcnt;
			}
			$grandqtrtotal+=$average;
			($average,$colorindex)=&Compute($qtrtotal[$n],$monthcnt);
			print "<TH BGCOLOR=\"$colors[$colorindex]\">$average</TH>";
		}
		($average,$colorindex)=&Compute($grandqtrtotal,$#Statements + 1);
		print "<TD>&nbsp;</TD><TH
BGCOLOR=\"$colors[$colorindex]\">$average</TH></TR>\n";
		if($newquarter==1) {
			$grandyrtotal=0;
			print "<TR><TH ALIGN=right>Year : </TH><TH
ALIGN=center>$rcyrtotal</TH>";
			foreach $n (0 .. $#Statements) {
				if($yrmonthcnt==0) {
					$average = 0;
				}else{
					$average = $yrtotal[$n] / $yrmonthcnt;
				}
				$grandyrtotal+=$average;
				($average,$colorindex)=&Compute($yrtotal[$n],$yrmonthcnt);
				print "<TH BGCOLOR=\"$colors[$colorindex]\">$average</TH>";
			}
			($average,$colorindex)=&Compute($grandyrtotal,$#Statements + 1);
			print "<TD>&nbsp;</TD><TH
BGCOLOR=\"$colors[$colorindex]\">$average</TH></TR>\n";
			foreach $n (0 .. $#Statements) {
				$yrtotal[$n]=0;
			}
			$rcyrtotal=0;
			$yrmonthcnt=0;
		}
		foreach $n (0 .. $#Statements) {
			$qtrtotal[$n]=0;
		}
		$rcqtrtotal=0;
		$monthcnt=0;
	}
  print <<EOF;
<TR><TD ALIGN=center>$month/$year</TD><TD ALIGN=center>$rcount{$key}</TD>
EOF
  $monthcnt++;
  $yrmonthcnt++;
  $oldquarter=$newquarter;
  $rctotal+=$rcount{$key};
  $rcqtrtotal+=$rcount{$key};
  $rcyrtotal+=$rcount{$key};
  $ltotal=0;
  foreach $i (0 .. $#Statements) {
    if($counts{$key}[$i]==0) {
      $average = 0;
    }else{
      $average = $results{$key}[$i] / $counts{$key}[$i];
    }
    $total[$i]+=$average;
    $qtrtotal[$i]+=$average;
    $yrtotal[$i]+=$average;
    $ltotal+=$average;
    ($average,$colorindex)=&Compute($results{$key}[$i],$counts{$key}[$i]);
    print "<TD ALIGN=center BGCOLOR=\"$colors[$colorindex]\">$average</TD>";
  }
  ($average,$colorindex)=&Compute($ltotal,$#Statements + 1);
  print "<TD>&nbsp;</TD><TH BGCOLOR=\"$colors[$colorindex]\">$average</TH></TR>";
}
$grandtotal=0;
print "<TR><TH ALIGN=right>Totals : </TH><TH ALIGN=center>$rctotal</TH>";
foreach $n (0 .. $#Statements) {
  if($#files== -1) {
    $average = $0;
  }else{
    $average = $total[$n] / ($#files + 1);
  }
  $grandtotal+=$average;
  ($average,$colorindex)=&Compute($total[$n],$#files + 1);
  print "<TH BGCOLOR=\"$colors[$colorindex]\">$average</TH>";
}
($average,$colorindex)=&Compute($grandtotal,$#Statements + 1);
print "<TD>&nbsp;</TD><TH BGCOLOR=\"$colors[$colorindex]\">$average</TH></TR>\n";

print "</TABLE>\n";

############################################################################

        open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
        while (<F>) {
                $footer .= $_;
        }
        close(F);

        print "$footer\n";
       &log("results","OK","-");
}

sub Compute {
  local($value,$count)=@_;
  local($average,$colorindex);

  if($count==0) {
    $average=0;
  }else{
    $average = $value / $count;
  }
  $average = sprintf("%.4f",$average);
  $average = sprintf("%.3f",$average);
  $average = sprintf("%.2f",$average);
  $average = sprintf("%.1f",$average);
  if($average == 5) {
    $colorindex=5;
  }elsif($average >= 4) {
    $colorindex=4;
  }elsif($average >= 3) {
    $colorindex=3;
  }elsif($average >= 2) {
    $colorindex=2;
  }elsif($average >= 1) {
    $colorindex=1;
  }else{
    $colorindex=0;
  }
  ($average,$colorindex);
}
1;
::::::::::::::
search.pl
::::::::::::::
########################################################################
# Method:       search
# Use:                  searches the NTRS MySQL database
# Created:      05/06/2002 MLN
########################################################################

sub search
{
       $keywords = $in{"keywords"};
       $boolean = $in{"boolean"};
       $mode = $in{"mode"};
       $debug = $in{"debug"};
       $limit = $in{"limit"};
       $offset = $in{"offset"};
       $method = $in{"method"};
       $total_hits = $in{"total_hits"};
       $url_title = $in{"title"};
       $url_creator = $in{"creator"};
       $url_date = $in{"date"};
       $url_type = $in{"type"};
       $url_description = $in{"description"};
       $orderby = $in{"orderby"};
       $order = $in{"order"};
       $archives = $in{"archives"};
       # if datestamp2 exists, it will override the value of datestamp
       $datestamp = $in{"datestamp"};
       $datestamp = $in{"datestamp2"} if ($in{"datestamp2"});

       my @fields = qw(title creator description date type);
       my ($header, $footer, $next_search, $now, $then, $elapsed);
```

```
# set the starting time
$then = time;

# set some defaults
$base = &MyBaseUrl;
$grey = "dedede";
$white = "white";
$orderby = "date" unless ($orderby);
$order = "" unless ($order eq "DESC");
$boolean = "AND" unless ($boolean);
undef($order) unless ($order eq "DESC");
# archives is separated by null's, since it is a repeated radio input
@archives = split(/\0/,$archives);

&http_header("text/html");

# get headers and footers

open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
while (<H>) {
        # remember which mode we were in
         if (/<!--$mode-->/) {
                 s/$grey/$white/g;
         }
        $header .= $_;
}
close(H);

print "$header\n";

open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
while (<F>) {
        $footer .= $_;
}
close(F);

use DBI;

$datasource = "DBI:mysql:database=ntrs;host=localhost";
$user = "mln";
$password = "ntrsr0cks";

#$dbh = DBI->connect($datasource, $user, $password, {'RaiseError' => 1});
$dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
        || db_error("$DBI::errstr");

# parse the keywords...
# take off any leading whitespace
$keywords =~ s/^\s+//g;
$keywords = &css_filter($keywords);

# split and rejoin for pretty printing
$print_keywords = $keywords;
$print_keywords =~ s/\+//g;
#$print_keywords =~ s/\"//g;
@keywords = split(/\s+/,$print_keywords);
$keywords = &add_and(@keywords);

print "<font color=red>mode = $mode</font>\n" if ($debug);

if ($mode eq "simple") {

        $boolean_mode = &boolean($keywords);

        # simple searches search only NASA archives

        $sql = "SELECT SQL_CALC_FOUND_ROWS oaiID, title, creator, \
        date, identifier, dateStamp, description, type, longName, \
        MATCH (title, creator, date, identifier, description, type) \
        AGAINST ('$keywords' $boolean_mode) AS score \
        FROM metadata, archives \
        WHERE ( (MATCH (title, creator, date, identifier, description, \
        type) AGAINST ('$keywords' $boolean_mode) ) AND \
        (metadata.archiveID = archives.archiveID) AND \
        (nasa = \"nasa\") ) ";

        print "<font color=red>keywords = -$keywords- </font>\n" if ($debug);
        print "<font color=red>sql = $sql</font><br>\n" if ($debug);

} elsif ($mode eq "advanced") {
```

```
               # first, process the keywords

               @title = split(/\s+/,$url_title);
               @creator = split(/\s+/,$url_creator);
               @date = split(/\s+/,$url_date);
               @type = split(/\s+/,$url_type);
               @description = split(/\s+/,$url_description);

               # hack for NACA
               # take out after the NACA report server is ready
               if ($keywords) {
                      push(@title,split(/\s+/,$keywords));
                      push(@creator,split(/\s+/,$keywords));
                      push(@date,split(/\s+/,$keywords));
                      push(@type,split(/\s+/,$keywords));
                      push(@description,split(/\s+/,$keywords));
               }

               push(@keywords,@title);
               push(@keywords,@creator);
               push(@keywords,@date);
               push(@keywords,@type);
               push(@keywords,@description);

               # these are all the keywords to search

               $keywords = &css_filter(join(" ",@keywords));
               $boolean_mode = &boolean($keywords);

               $print_keywords .= "$url_title (title) $boolean " if ($url_title);
               $print_keywords .= "$url_creator (author) $boolean " if ($url_creator);
               $print_keywords .= "$url_date (date) $boolean " if ($url_date);
               $print_keywords .= "$url_type (citation) $boolean " if ($url_type);
               $print_keywords .= "$url_description (abstract) $boolean " if
($url_description);
               # remove the space + trailing boolean + space
               $print_keywords =~ s/\s$boolean\s+$//;

               # build the fielded full-text statement

               $sql = "SELECT SQL_CALC_FOUND_ROWS oaiID, title, creator, \
               date, identifier, dateStamp, description, type, longName, \
               shortName FROM metadata, archives \
               WHERE ( (metadata.archiveID = archives.archiveID) AND (";

               if ($url_title) {
                      $url_title = &add_and(@title);
                      $sql .= "(MATCH (title) AGAINST ('$url_title' $boolean_mode))
$boolean";
               }
               if ($url_creator) {
                      $url_creator = &add_and(@creator);
                      $sql .= "(MATCH (creator) AGAINST ('$url_creator' $boolean_mode))
$boolean";
               }
               if ($url_date) {
                      $url_date = &add_and(@date);
                      $sql .= "(MATCH (date) AGAINST ('$url_date' $boolean_mode))
$boolean";
               }
               if ($url_type) {
                      $url_type = &add_and(@type);
                      $sql .= "(MATCH (type) AGAINST ('$url_type' $boolean_mode))
$boolean";
               }
               if ($url_description) {
                      $url_description = &add_and(@description);
                      $sql .= "(MATCH (description) AGAINST ('$url_description'
$boolean_mode)) $boolean";
               }

               # which archives to search?
               # the archives get OR'd -- the hits can come from any archive

               #$sql .= " AND (";
               $sql =~ s/$boolean$/) AND (/;

               # if no fields have been entered, rig it to return nothing
                # instead of everthing (which is very expensive)
                if ( (!$url_title) && (!$url_creator) && (!$url_date) &&
                     (!$url_type) && (!$url_description) ) {
                        $sql .= "MATCH (title, creator, date, identifier, \
                                description, type) AGAINST ('$keywords' \
                                $boolean_mode)) AND( ";
                }

               foreach $a (@archives) {
                      print "<font color=red>archive = $a </font>" if ($debug);
                      $sql .= " (shortName = \"$a\") OR ";
               }
               $sql =~ s/OR\ $/))/g;

               if ($debug) { print "<font color=red>$sql</font><br>\n"; }

       } elsif ($mode eq "browse") {
               $sql = "SELECT SQL_CALC_FOUND_ROWS oaiID, title, creator, \
               date, identifier,  dateStamp, description, type \
               FROM metadata WHERE ";

               # which archives to search? -- just the first one since
               # this is a browse

               print "<font color=red>archive = $archives[0] </font><br>" if ($debug);

               # figure out what the long name is right now
               # also grab $total_hits to avoid the double searching
               # for browse
               $sql2 = "SELECT longName, archiveID, numRecords FROM \
                      archives WHERE ( shortName = \"$archives[0]\" ) ";
               $sth = $dbh->prepare($sql2);
               $sth->execute;
               ($longName, $archiveID, $total_hits) = $sth->fetchrow_array;
               $print_keywords = "all records in $longName";

               $browse_longname = $longName;

               $sql .= " archiveID = $archiveID ";

               print "<font color=red>$sql</font><br>\n" if ($debug);

       } elsif ($mode eq "updates") {
               $sql = "SELECT SQL_CALC_FOUND_ROWS oaiID, title, creator, \
               date, identifier, dateStamp, description, type, longName \
       FROM metadata, archives\
               WHERE ( (metadata.archiveID = archives.archiveID) AND \
               (dateStamp >= \"$datestamp\") ";

               $print_keywords = "all records ";
               if ($archives[0]) {
                      $sql .= "AND (archives.shortName = \"$archives[0]\") ";

                      # figure out what the long name is right now
                      $sql2 = "SELECT longName, archiveID FROM \
                      archives WHERE ( shortName = \"$archives[0]\" ) ";
                      $sth2 = $dbh->prepare($sql2);
                      $sth2->execute;
                      ($longName, $archiveID) = $sth2->fetchrow_array;
                      $sth2->finish;
                      $print_keywords .= "from $longName ";
               }

               $sql .= " ) ";

               $print_keywords .= "added or updated since $datestamp";

               if ($debug) { print "<font color=red>$sql</font><br>\n"; }
       } else {
               die "error, unknown mode = \"$mode\"\n";

       }

       # below this point is the generic searching function, independent
       # of the mode (simple, advanced, browse)

       if ( ($boolean_mode) || ($mode eq "browse") ) {
               $sql .= " ORDER BY $orderby $order ";
       }
       $sql .= " LIMIT $offset, $limit";

       print "<font color=red>sql = $sql </font><br>\n" if ($debug);
```

```
# now do the real search...
$sth = $dbh->prepare($sql);
$sth->execute;

# if its our first search, we need to know how many total hits
# would have been returned if we did not use LIMIT
# if we already know the offset, we have total_hits, and we can
# skip this part

if ( $offset == 0 ) {
        $sth2 = $dbh->prepare("SELECT FOUND_ROWS()");
        $sth2->execute;
        ($total_hits) = $sth2->fetchrow_array;
        $sth2->finish;
}

$count = $offset + 1;
$hits = $sth->rows();

# first pagination buttons
&pagination($print_keywords,$offset,$limit,$hits,$total_hits);

print "<table cellpadding=5 cellspacing=0 border=0 width=100%>";
while (($oaiID,$title,$creator,$date,$identifier, $datestamp,
        $description,$type,$longName) = $sth->fetchrow_array ) {

        # bold the results
        # do it this way to preserve the original case
        if ($boolean_mode) {
                @keywords = &strip_boolean(@keywords);
        }
        foreach $k (@keywords) {
                # names and such can have ".", which is confusing
                # for the parser below.  ignore rather than quote
                $k =~ s/\.//g;

                # also skip middle initials...
                # must think of a better way to do this
                next if ($k =~ /^.$/);

                # also "or" and "and"
                # must think of a better way to do this too
                next if ($k =~ /^or$/i);
                next if ($k =~ /^and$/i);

                # also strip any double quotes -- they're for MySQL,
                # not for highlighting
                $k =~ s/\"//g;

                if ($title =~ /$k/i) {
                        $title =~ s#$&#<strong>$&</strong>#ig;
                }
                if ($creator =~ /$k/i) {
                        $creator =~ s#$&#<strong>$&</strong>#ig;
                }
                if ($longName =~ /$k/i) {
                        $longName =~ s#$&#<strong>$&</strong>#ig;
                }
                if ($description =~ /$k/i) {
                        $description =~ s#$&#<strong>$&</strong>#ig;
                }
                if ($date =~ /$k/i) {
                        $date =~ s#$&#<strong>$&</strong>#ig;
                }
                if ($type =~ /$k/i) {
                        $type =~ s#$&#<strong>$&</strong>#ig;
                }
        }

        # change color every other row
        if ( ($count % 2) == 0 ) {
                $color = $grey;
        } else {
                $color = $white;
        }

        # identifiers could have been concatenated
        # split, hyperlink, rejoin
        # note the extra space on the split
        @identifiers = split("; ",$identifier);
```

```
		foreach (@identifiers) {
			# only hyperlink http:// or ftp:// identifiers
			next unless (/(ht|f)tp:\/\//);
			# some DOIs have "<" and ">" chars in them!
			s/\s*//g;
			# semi-colons in the URL are not "split" on and
			# must be encoded (ref: some GISS DOIs)
			s/;/%3B/g;
			s/.*/<a href=\"?method=display&redirect=$&&oaiID=$oaiID\">$&<\/a>/;
		}

		if (!$identifier) {
			$identifier = "<a href=?method=ordering&oaiID=$oaiID>No Digital
Version Available - Order This Document</a>";
		} else {
			$recommendation = "<a href=\"#\"
onClick=\"window.open(\'?method=recommendation&oaiID=$oaiID\',\'recommendations\',\'locat
ion=no,menubar=no,status=no,toolbar=no,scrollbars=yes\'); return false;\">Recommendations
for Related Documents</a>";
			#$recommendation = "<a href=\"#\"
onClick=\"window.open(\'?method=recommendation&oaiID=$oaiID\',\'recommendations\',\'locat
ion=no,menubar=no,status=no,toolbar=no,scrollbars=yes\'); return false;\"><img
src=http://ntrs.nasa.gov/images/z2.gif alt=\"Zeitgeist logo\">Recommendations for Related
Documents</a>";
			push (@identifiers,"$recommendation");
			$identifier = join(" <br>\n ",@identifiers);
		}


		# longName doesn't get set in browse mode...
		$longName = $browse_longname if (!$longName) ;

		# remove this after the OK from JoAnne
		if ( ($longName =~ /^NASA Ames/) ||
		  ($longName =~ /^NASA Stennis/) ||
		  ($longName =~ /^NASA Goddard Space/) ||
		  ($longName =~ /^NASA Kennedy/) ) {
			undef($identifier);
		}

		print <<EOF;
<tr><td bgcolor=$color valign=top>$count.</td>
<td bgcolor=$color>$title<font size=-1><i><br>$creator<br>$longName<br>$type
EOF
	if ($type =~ /$date/) {
		# date is part of type, so print nothing
	} elsif (($type) && ($date)) {
		print ", $date";
	} elsif ($date) {
		print "$date";
	}

		print <<EOF;
</i>
<br>$description
<br>$identifier
<br>Updated/Added to NTRS: $datestamp
</font></td>
EOF
		$count++;
	} # end of while()


	# print a mesg if no hits were found
	print "<tr><td align=center>no search results were found\n" if ($hits == 0);

	print "</table><p>\n";

	# pagination only if there were hits
	&pagination($print_keywords,$offset,$limit,$hits,$total_hits) if ($hits);

	$sth->finish;
	$dbh->disconnect;
	$now = time;
	$elapsed = $now - $then;

	# visible if debugging, in the comments otherwise
	if ($debug) {
		print "<font color=red>total seconds = $elapsed</font>\n";
	} else {
		print "<!-- total seconds = $elapsed -->\n";
```

```
        }

        print "$footer\n";

        &log("search","OK","keywords=$keywords limit=$limit offset=$offset
boolean=$boolean");
}

sub pagination {
        my($print_keywords,$offset,$limit,$hits,$total_hits) = @_;
        my($start,$end,$newoffset,$args);

        $args .= "method=$method";
        $args .= "&mode=$mode";
        $args .= "&boolean=$boolean";
        $args .= "&debug=$debug" if ($debug);
        $args .= "&datestamp=$datestamp" if ($datestamp);
        $args .= "&order=$order" if ($order);
        $args .= "&orderby=$orderby" if ($orderby);
        $args .= "&datestamp=$datestamp" if ($datestamp);
        $args .= "&datestamp2=$datestamp2" if ($datestamp2);
        $args .= "&keywords=$keywords" if (($keywords) && ($mode eq "simple"));
        $args .= "&title=$url_title" if ($url_title);
        $args .= "&creator=$url_creator" if ($url_creator);
        $args .= "&date=$url_date" if ($url_date);
        $args .= "&type=$url_type" if ($url_type);
        $args .= "&description=$url_description" if ($url_description);
        $args .= "&limit=$limit";
        $args .= "&hits=$hits";
        $args .= "&total_hits=$total_hits";
        foreach $a (@archives) {
                $args .= "&archives=$a";
        }

        # add +'s back in so we can build the "next", "previous" etc. urls
        $args =~ s/\+/%2B/g;
        $args =~ s/\s/+/g;

        # now build the string to pass on for the next URL
        # this will have everything but the new offset

        # pagination buttons
        print "<table cellpadding=5 border=0>\n";

        print "<td bgcolor=$grey align=center>searching for \"$print_keywords\"\n";
        # calculate ranges
        $start = $offset + 1;
        $end = $offset + $hits;
        # avoid things like "displaying 1 - 0 of a total of 0"
        if ($hits == 0) { $start = 0; }
        print "<td bgcolor=$grey align=center>displaying records $start - $end of a total
of $total_hits \n";

        if ($offset == 0) {
                # no previous button
        } else {
                # first button
                print "<td bgcolor=$grey>\n";
                print "<a href=$base?$args&offset=0>first $limit</a>\n";

                # previous button
                $newoffset = $offset - $limit;
                if ($newoffset >= 0) {
                        print "<td bgcolor=$grey>\n";
                        print "<a href=$base?$args&offset=$newoffset>previous
$limit</a>\n";
                }
        }

        if ($hits == $limit) {
                # next button
                $newoffset = $offset + $limit;
                if ($newoffset != $total_hits) {
                        print "<td bgcolor=$grey>\n";
                        print "<a href=$base?$args&offset=$newoffset>next $limit</a>\n";
                }

                # last button
                print "<td bgcolor=$grey>\n";
                $newoffset = $total_hits - $limit;
                print "<a href=$base?$args&offset=$newoffset>last $limit</a>\n";
        } else {
                # no next button
        }
        print "</table>\n";
}

sub boolean {
        my($keywords) = @_;

        # do the keywords have MySQL boolean operators?

        #if ($keywords =~ /[+-<>~*"]/) {
                # do something clever with $dbh->quote();  ????
                return("IN BOOLEAN MODE");
        #} else {
                #return(undef($boolean_mode));
        #}

}

sub strip_boolean {
        my(@words) = @_;
        my(@newwords);

        foreach $w (@words) {
                next if ($w =~ /^-/);  # ignore negated words
                #$w =~ s/[\+\-\<\>\~\*"]//g;
                # took the dash (-) out so hyphenated words would be
                # correctly displayed
                $w =~ s/[\+\<\>\~\*"]//g;
                push (@newwords,$w);
        }
        return(@newwords);
}

sub add_and {
        my(@keywords) = @_;
        my ($k, @newkeywords,$test_for_quotes);

        #print "keywords = @keywords\n" if ($debug);

        $test_for_quotes = join(" ",@keywords);

        if ($test_for_quotes =~ /".*"/) {
                return (join(" ",@keywords));
        }

        foreach $k (@keywords) {
                $k = "+$k" unless ($k =~ /\+\-\"/);
                push(@newkeywords,$k);
        }
        return(join(" ",@newkeywords));

}

# to remove the cross-site scripting vulnerability
# from:  http://www.cert.org/tech_tips/malicious_code_mitigation.html/
# CERT Advisory CA-2000-02

sub css_filter {
         my($fd) = @_;
         #$fd =~ s/[\<\>\"\'\%\;\)\(\&\+]//g;
        # I removed the double quote (") so we can have quoted searching
         $fd =~ s/[\<\>\'\%\;\)\(\&\+]//g;
         return( $fd ) ;
}

1;
::::::::::::::
simple.pl
::::::::::::::
#########################################################################
# Method:       simple
# Use:                  simple search for NTRS
# Created:      05/06/2002 MLN
#########################################################################

sub simple
{
         &http_header("text/html");
        my($base) = &MyBaseUrl;
        # get headers and footers

        open(H,"ntrs.pkg/header.html") || die ("$!: cannot open header\n");
        while (<H>) {
               # turn on the "simple search" button
               if (/<!--simple-->/) {
                      s/dedede/white/g;
               }
                $header .= $_;
        }
        close(H);

        print "$header\n";

        open(F,"ntrs.pkg/footer.html") || die ("$!: cannot open footer\n");
        while (<F>) {
                $footer .= $_;
        }
        close(F);

        &news;

        print <<EOF;
<table cellspacing=10>
<tr>
<td align=center>
<form action=$base method=GET>
<input type=hidden name=method value=search>
<input type=hidden name=limit value=25>
<input type=hidden name=offset value=0>
<input type=hidden name=mode value=simple>
<input type=hidden name=order value=DESC>


<label for=keywords> </label>
<textarea name="keywords" cols=40 rows=4 id=keywords></textarea>

<tr>
<td bgcolor=white align=center>
<input type="submit" label="Begin Search" value="Begin Search">
<input type="reset" label="Reset Search" value="Reset Search"><p>

</form>

</table>
<p>
EOF

        print "$footer\n";
        &log("simple","OK","-");
}

sub news {

        open(N,"ntrs.pkg/news.rss") || die ("$!: cannot open news\n");
        while (<N>) {
                push(@news,$_);
        }
        close(N);

        print "<table>\n";
        #print "<tr><td align=center><font size=-1><b>News:</b></font>\n";
        # process the RSS file
        undef($title);
        undef($item);
        foreach $line (@news) {

               if ($title_num == 1) {
                      print "</table>\n";
                      return;
               }

                next unless (($line =~ /<item>/) || ($item));
                $item=1;

                if ($line =~ /<title>/) {
                        $title .= $line;
                } elsif ($title) {
                        $title .= $line;
                }
                if ($line =~ /<\/title>/) {
                        $title_num++;
                        # print $title;
                        $title =~ s/<title>//g;
                        $title =~ s#</title>##g;
                       if ($title_num == 1) {
                              print "<tr bgcolor=dedede><td align=center><font size=-
1>Latest News: $title <a href=?method=news>(more news...)</a></font>\n";
                       } elsif ($title_num % 2) {
                                print "<tr bgcolor=dedede><td><font
size=1>$title</font>\n";
                        } else {
                                print "<tr><td><font size=1>$title</font>\n";
                        }
                        undef($title);
                        undef($item);
                }
        }
       print "<tr><td align=center><font size=-1><a href=?method=news>(more
news...)</a></font>\n";
        print "</table>\n";

}

1;
```

Appendix 5: NTRS Source Code of bin Scripts

```
::::::::::::::
bin/archives-html.pl
::::::::::::::
#!/usr/local/bin/perl

use DBI;

$datasource = "DBI:mysql:database=ntrs;host=localhost";
$user = "mln";
$password = "ntrsr0cks";

$dir = "/usr/local/web/htdocs/ntrs/ntrs/ntrs.pkg/";

$dbh = DBI->connect($datasource, $user, $password, {'RaiseError' => 1});


# get the nasa archives...

$sql = "SELECT shortName, longName FROM archives WHERE nasa = 'nasa' \
        ORDER BY longName ASC";
$sth = $dbh->prepare($sql);
$sth->execute;

$i = 0;
while ( ($nasa_archives[$i]{"shortName"}, $nasa_archives[$i]{"longName"}) =
        $sth->fetchrow_array) {
        $i++;
}
$sth->finish;

# get the non-nasa archives...

$sql = "SELECT shortName, longName FROM archives WHERE nasa = 'non-nasa' \
        ORDER BY longName ASC";
$sth = $dbh->prepare($sql);
$sth->execute;

$i = 0;
while (($nonnasa_archives[$i]{"shortName"}, $nonnasa_archives[$i]{"longName"}) =
        $sth->fetchrow_array) {
        $i++;
}
$sth->finish;

# write the html...

foreach $type ("checkbox", "radio") {

        print "writing $dir$type.html\n";

        if ($type eq "checkbox") {
                $verb = "search";
        } else {
                $verb = "browse";
        }

        $i = 0;
        open(F,">$dir$type.html") || die "cannot open $dir$type.html - $!";
        print F "<table summary=\"a list of all NASA and Non-NASA archives\">\n" .
        "<tr>\n" .
        "<td colspan=2 align=center>Select the databases you would like to $verb\n" .
        "<tr>\n" .
        "<th align=center bgcolor=dedede> NASA Archives</th>\n" .
        "<th align=center bgcolor=dedede> Non-NASA Archives</th>";

        while ( ($nonnasa_archives[$i]{"shortName"}) ||
                ($nasa_archives[$i]{"shortName"}) ) {
                print F "<tr>\n";
                if ($nasa_archives[$i]{"shortName"}) {
                        $out = "<td> <input type=\"$type\" name=\"archives\"
value=\"$nasa_archives[$i]{'shortName'}\" id=\"$nasa_archives[$i]{'shortName'}\"> <label
for=\"$nasa_archives[$i]{'shortName'}\">$nasa_archives[$i]{'longName'}</label>\n";
                        # NASA archives are checked by default
                        # or, if this is radio, LaRC is checked by default ;-)
                        if ( ($type eq "checkbox") ||
                        ($out =~ /ltrs.larc.nasa.gov/) ) {
                          $out =~ s/id=/CHECKED id=/;
                   }
                   print F "$out";
            } else {
                   print F "<td> &nbsp;\n";
            }
            if ($nonnasa_archives[$i]{"shortName"}) {
                   print F "<td> <input type=\"$type\" name=\"archives\"
value=\"$nonnasa_archives[$i]{'shortName'}\" id=\"$nonnasa_archives[$i]{'shortName'}\">
<label
for=\"$nonnasa_archives[$i]{'shortName'}\">$nonnasa_archives[$i]{'longName'}</label>\n";
            } else {
                   print F "<td> &nbsp;\n";
            }

            $i++;
      }

      print F "</table>\n";
      close (F);
}

#
# print out the pull down menu (select)
# this is significantly different from the table generation above
# so it is handled separately
#

$type = "select";
print "writing $dir$type.html\n";
open(F,">$dir$type.html") || die "cannot open $dir$type.html - $!";
print F "<select name=\"archives\" id=\"archives\">\n";
print F "<option selected value=\"\"> All Archives\n";
$i=0;
while ($nasa_archives[$i]{"shortName"}) {

      if ($nasa_archives[$i]{"shortName"}) {
            print F "<option
value=$nasa_archives[$i]{'shortName'}>$nasa_archives[$i]{'longName'}\n";
      }
      $i++;
}

$i=0;
while ($nonnasa_archives[$i]{"shortName"}) {
      if ($nonnasa_archives[$i]{"shortName"}) {
            print F "<option
value=$nonnasa_archives[$i]{'shortName'}>$nonnasa_archives[$i]{'longName'}\n";
      }
      $i++;
}
print F "</select><br>\n";
close (F);

print "finished\n";

$dbh->disconnect;

exit;

::::::::::::::
bin/build-table.pl
::::::::::::::
#!/usr/local/bin/perl


use DBI;

$datasource = "DBI:mysql:database=ntrs;host=localhost";
$user = "mln";
$password = "ntrsr0cks";

@dc_tags = ("title", "creator", "subject", "description", "publisher",
            "contributor", "date", "type", "format", "identifier",
            "source", "language", "relation", "coverage", "rights");

$dbh = DBI->connect($datasource, $user, $password, {'RaiseError' => 1});

$sql = "drop table if exists harvests";
$dbh->do($sql);
```

```
$sql = "create table harvests (day date not null, archiveID int not null, \
size int not null, PRIMARY KEY (day,archiveID,size) )";
$dbh->do($sql);

$sql = "drop table if exists archives";
$dbh->do($sql);
$sql = "create table archives (archiveID int not null, shortName text, \
longName text, nasa text, url text, numRecords int, primary key (archiveID))";
$dbh->do($sql);

$sql = "drop table if exists metadata";
$dbh->do($sql);
$sql = "create table metadata (oaiID text not null, \
dateStamp date not null, archiveID int not null, title text, creator text, \
subject text, description text, publisher text, contributor text, date text, \
type text, format text, identifier text, source text, language text, \
relation text, coverage text, rights text, FULLTEXT (title, creator, date, \
identifier, description, type), FULLTEXT(title), FULLTEXT (creator), \
FULLTEXT (date), FULLTEXT (type), FULLTEXT (description), INDEX (archiveID), \
INDEX (dateStamp), primary key (oaiID(255)) )";
$dbh->do($sql);


$dbh->disconnect;

exit;

::::::::::::::
bin/delete.pl
::::::::::::::
#!/usr/bin/perl
#
# usage:       delete.pl {database|filedatabase} id
#

use DBI;

$datasource = "DBI:mysql:database=ntrs;host=localhost";
$user = "mln";
$password = "ntrsr0cks";

$metadata_dir = "/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/";

$option = $ARGV[0];
$id = $ARGV[1];

die "an identifier needs to be supplied!\n" unless ($id);

#
# deleting from the database is always done
#

$dbh = DBI->connect($datasource, $user, $password, {'RaiseError' => 1});

$sql = "DELETE FROM metadata WHERE oaiID like \"$id\"";
$rows = $dbh->do($sql);

if ($rows == 1) {
        print "$rows record deleted from the database\n";
} else {
        $rows = 0 if ($rows eq "0E0");
        print "$rows records deleted from the database\n";
}

$dbh->disconnect;

#
# deleting from the filesystem is optional
#

if ($option eq "filedatabase") {

        (undef,$archive,undef) = split(/:/,$id);

        $deleted = unlink("$metadata_dir/$archive/metadata/$id");

        if ($deleted > 0) {
                print "deleted $id ($metadata_dir/$archive/metadata/$id)\n";
        } else {
                print "could not delete $id ($metadata_dir/$archive/metadata/$id)\n";
        }
}

exit;

::::::::::::::
bin/entities.pl
::::::::::::::
#!/usr/bin/perl

%entities = (
        "amp"  =>      "&",
        "gt"   =>      ">",
        "lt"   =>      "<",
        "apos" =>      "\'",
        "quot" =>      "\""
);

sub remove_entities {
        my($string) = @_;

        foreach $e (keys(%entities)) {
                $string =~ s/\&$e;/$entities{"$e"}/g;
        }

        return($string);

}

::::::::::::::
bin/harvest-cron
::::::::::::::
#!/usr/local/bin/perl


#umask 000

use DBI;

$datasource = "DBI:mysql:database=ntrs;host=localhost";
$user = "mln";
$password = "ntrsr0cks";

$today = `/usr/local/web/htdocs/ntrs/bin/today.pl`;
$populate = "/usr/local/web/htdocs/ntrs/bin/populate-ntrs.pl";
$harvest_dir = "/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/";
$log = "/usr/local/web/htdocs/ntrs/ntrs/_log.pkg/harvests/$today";

open (F,">$log") || die "cannot open $log : $!\n";

$date = `date`;
print F "$date\n";
print "$date\n";

$dbh = DBI->connect($datasource, $user, $password, {'RaiseError' => 1});

$sql = "SELECT shortName FROM archives";
$sth = $dbh->prepare("$sql");
$sth->execute;
while ( ($shortName) = $sth->fetchrow_array) {

        print "harvesting $shortName\n";
        print F "harvesting $shortName\n";
        $output = `$harvest_dir$shortName/harvest.pl`;
        print F "$output\n";
        print "$output\n";
}
$sth->finish;


print "populating...\n";
print F "populating...\n";
$output = `$populate`;
print F "$output\n";
print "$output\n";

unlink ("/usr/local/web/htdocs/ntrs/ntrs/oai/id.db");

$date = `date`;
print F "$date\n";
print F "$date\n";
print "$date\n";

close (F);
$dbh->disconnect;


::::::::::::::
bin/harvest-cron.old
::::::::::::::
#!/bin/csh -x

umask 000

set today = `/usr/local/web/htdocs/ntrs/bin/today.pl`;
set log = /usr/local/web/htdocs/ntrs/ntrs/_log.pkg/harvests/$today

date >>& $log


cd /usr/local/web/htdocs/ntrs/ODL-Harvest-2.0/Harvest/

arxiv.org/harvest.pl >>& $log
ecd.osti.gov/harvest.pl >>& $log
biomedcentral.com/harvest.pl >>& $log
ltrs.larc.nasa.gov/harvest.pl  >>& $log
naca.larc.nasa.gov/harvest.pl >>& $log
www.giss.nasa.gov/harvest.pl >>& $log
genesis.jpl.nasa.gov/harvest.pl >>& $log

cd /usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/
harvest-all >>& $log

cd /usr/local/web/htdocs/ntrs/bin
populate-ntrs.pl >>& $log
rm -f /usr/local/web/htdocs/ntrs/ntrs/oai/id.db
date >>& $log

::::::::::::::
bin/log-cron
::::::::::::::
#!/usr/local/bin/perl
#
# rotate log files, process awstats

use File::Copy;
use Shell qw (gzip grep);

$today = `/usr/local/web/htdocs/ntrs/bin/today.pl`;
$usage = "/usr/local/web/htdocs/ntrs/bin/usage.pl";
$http_log = "/var/web/ntrs/access";
$http_tmp_log = "/var/web/ntrs/access.tmp";
$http_log_archive = "/usr/local/web/htdocs/ntrs/ntrs/_log.pkg/http/";

%months = (     "01",   "Jan",
                "02",   "Feb",
                "03",   "Mar",
                "04",   "Apr",
                "05",   "May",
                "06",   "Jun",
                "07",   "Jul",
                "08",   "Aug",
                "09",   "Sep",
                "10",   "Oct",
                "11",   "Nov",
                "12",   "Dec"
);

($year,$month,$day) = split (/-/,$today);

if ( ($month--) == 0 ) {
        $year--;
        $month = 12;
} else {
        if ($month < 10) {
                $month = "0$month";
        }
}

#
# rotate bucket log
#
```

```
chdir ("/usr/local/web/htdocs/ntrs/ntrs/_log.pkg");

move ("access.log", "access/access.$year-$month.log");
open (F,">access.log");
close (F);
chmod(0666,"access.log");

gzip("access/access.$year-$month.log");

#
# run awstats
#

mkdir ("/usr/local/web/htdocs/ntrs/ntrs/_log.pkg/awstats/$year-$month",0777);

$result = `/usr/local/web/htdocs/ntrs/awstats-5.4/tools/awstats_buildstaticpages.pl -
config=ntrs.nasa.gov --awstatsprog=/usr/local/web/htdocs/ntrs/awstats-5.4/wwwroot/cgi-
bin/awstats.pl -month=$month -year=$year -update -
dir=/usr/local/web/htdocs/ntrs/ntrs/_log.pkg/awstats/$year-$month`;

print "$result\n";

# make an index.html to avoid the directory listing

`ln -s /usr/local/web/htdocs/ntrs/ntrs/_log.pkg/awstats/$year-
$month/awstats.ntrs.nasa.gov.html /usr/local/web/htdocs/ntrs/ntrs/_log.pkg/awstats/$year-
$month/index.html`;

#
# process the http log
#

$date = "/" . $months{$month} . "/$year";

`grep $date $http_log > $http_log_archive$year-$month`;

# now reset the current access log
($thisyear,$thismonth,$thisday) = split (/-/,$today);
$date = "/" . $months{$thismonth} . "/$thisyear";

#`grep $date $http_log > $http_tmp_log`;

# keep an old copy just in case

#rename ("$http_log", "$http_log.bak");

# reset the temp file to the real access log

#rename ("$http_tmp_log", "$http_log");

# now gzip the back up copy at our leisure

#gzip("$http_log.bak");

# process the bucket logs into day logs for the analyze method

`cat $http_log_archive$year-$month | $usage`;

# gzip the bucket log for storage

gzip("$http_log_archive$year-$month");
::::::::::::::
bin/populate-ntrs.pl
::::::::::::::
#!/usr/local/bin/perl

use DBI;

use lib '/usr/local/web/htdocs/ntrs/ODL-Harvest/lib';
use Pure::EZXML;
use Pure::EZHTTP;
use Pure::X2D;

$today = "/usr/local/web/htdocs/ntrs/bin/today.pl";

$datasource = "DBI:mysql:database=ntrs;host=localhost";
$user = "mln";
$password = "ntrsr0cks";

$metadata_root = "/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/";
```

```
$metadata_tail = "/metadata";

@dc_tags = qw(title creator subject description publisher
              contributor date type format identifier
              source language relation coverage rights);

$archives{"naca.larc.nasa.gov"}{"id"} = "1";
$archives{"naca.larc.nasa.gov"}{"long_name"} = "National Advisory Committee for
Aeronautics (NACA)";
$archives{"naca.larc.nasa.gov"}{"url"} = "http://naca.larc.nasa.gov/";
$archives{"naca.larc.nasa.gov"}{"nasa"} = "nasa";

$archives{"ltrs.larc.nasa.gov"}{"id"} = "2";
$archives{"ltrs.larc.nasa.gov"}{"long_name"} = "NASA Langley Research Center";
$archives{"ltrs.larc.nasa.gov"}{"url"} = "http://techreports.larc.nasa.gov/ltrs/";
$archives{"ltrs.larc.nasa.gov"}{"nasa"} = "nasa";

$archives{"jtrs"}{"id"} = "3";
$archives{"jtrs"}{"long_name"} = "NASA Johnson Space Center";
$archives{"jtrs"}{"url"} = "http://ston.jsc.nasa.gov/collections/TRS/";
$archives{"jtrs"}{"nasa"} = "nasa";

# icase was id #4

$archives{"riacs"}{"id"} = "5";
$archives{"riacs"}{"long_name"} = "RIACS (NASA Ames Research Center)";
$archives{"riacs"}{"url"} = "http://eprints.riacs.edu/";
$archives{"riacs"}{"nasa"} = "nasa";

$archives{"arc"}{"id"} = "6";
$archives{"arc"}{"long_name"} = "Aeronautical Research Council (UK)";
$archives{"arc"}{"url"} = "http://www.magic.ac.uk/";
$archives{"arc"}{"nasa"} = "non-nasa";

$archives{"atrs"}{"id"} = "7";
$archives{"atrs"}{"long_name"} = "NASA Ames Research Center";
$archives{"atrs"}{"url"} = "";
$archives{"atrs"}{"nasa"} = "nasa";

$archives{"ktrs"}{"id"} = "8";
$archives{"ktrs"}{"long_name"} = "NASA Kennedy Space Center";
$archives{"ktrs"}{"url"} = "";
$archives{"ktrs"}{"nasa"} = "nasa";

$archives{"ssctrs"}{"id"} = "9";
$archives{"ssctrs"}{"long_name"} = "NASA Stennis Space Center";
$archives{"ssctrs"}{"url"} = "";
$archives{"ssctrs"}{"nasa"} = "nasa";

$archives{"gtrs"}{"id"} = "10";
$archives{"gtrs"}{"long_name"} = "NASA Goddard Space Flight Center";
$archives{"gtrs"}{"url"} = "";
$archives{"gtrs"}{"nasa"} = "nasa";

$archives{"mtrs"}{"id"} = "11";
$archives{"mtrs"}{"long_name"} = "NASA Marshall Space Flight Center";
$archives{"mtrs"}{"url"} = "http://trs.msfc.nasa.gov/";
$archives{"mtrs"}{"nasa"} = "nasa";

$archives{"arxiv.org"}{"id"} = "12";
$archives{"arxiv.org"}{"long_name"} = "arXiv Physics Eprint Server";
$archives{"arxiv.org"}{"url"} = "http://www.arxiv.org/";
$archives{"arxiv.org"}{"nasa"} = "non-nasa";

$archives{"casi"}{"id"} = "13";
$archives{"casi"}{"long_name"} = "NASA Center for AeroSpace Information (CASI)";
$archives{"casi"}{"url"} = "http://www.sti.nasa.gov/";
$archives{"casi"}{"nasa"} = "nasa";

$archives{"ecd.osti.gov"}{"id"} = "14";
$archives{"ecd.osti.gov"}{"long_name"} = "Energy Citation Database (OSTI)";
$archives{"ecd.osti.gov"}{"url"} = "http://www.osti.gov/energycitations/";
$archives{"ecd.osti.gov"}{"nasa"} = "non-nasa";

$archives{"biomedcentral.com"}{"id"} = "15";
$archives{"biomedcentral.com"}{"long_name"} = "BioMed Central";
$archives{"biomedcentral.com"}{"url"} = "http://www.biomedcentral.com/";
$archives{"biomedcentral.com"}{"nasa"} = "non-nasa";

$archives{"genesis.jpl.nasa.gov"}{"id"} = "16";
```

```
$archives{"genesis.jpl.nasa.gov"}{"long_name"} = "GENESIS (NASA Jet Propulsion
Laboratory)";
$archives{"genesis.jpl.nasa.gov"}{"url"} = "http://genesis2.jpl.nasa.gov/";
$archives{"genesis.jpl.nasa.gov"}{"nasa"} = "nasa";

$archives{"www.giss.nasa.gov"}{"id"} = "17";
$archives{"www.giss.nasa.gov"}{"long_name"} = "NASA Goddard Institute for Space Studies";
$archives{"www.giss.nasa.gov"}{"url"} = "http://www.giss.nasa.gov/gpol/";
$archives{"www.giss.nasa.gov"}{"nasa"} = "nasa";

$parser = new Pure::EZXML;

$dbh = DBI->connect($datasource, $user, $password, {'PrintError' => 0});

&build_archive_table;

if (@ARGV) {
        @a = @ARGV;

} else {
        @a = keys(%archives);
}

foreach $a (@a) {
        print "building $a\n";
        &build_metadata_table("$a");
        print "finding the size of $a\n";
        $sql = "SELECT COUNT(title) FROM metadata WHERE archiveID=$archives{$a}{'id'}";
        $sth = $dbh->prepare("$sql");
        $sth->execute || die "could not count\n";
        ($size) = $sth->fetchrow_array;
        $sth->finish;
        print "size = $size sql = $sql\n";
        $sth = $dbh->prepare("UPDATE archives SET numRecords=$size \
                where archiveID='$archives{$a}{'id'}' ");
        $sth->execute;
        $sth->finish;
        $day = `$today`;
print "day = $day\n";
        $sql = "INSERT IGNORE INTO harvests (day, archiveID, size)
VALUES(\"$day\",$archives{$a}{'id'},$size)";
        $dbh->do("$sql") || die "could not update harvest: $!\n";
};

$dbh->disconnect;

exit;

###############################################################################
# build_archive_table
###############################################################################

sub build_archive_table {
        my $sql;

        $sql = "explain archives";
        $sth = $dbh->prepare($sql);
        $sth->execute || return;
        $sth->finish;

        foreach $a (keys(%archives)) {

$sql = <<EOF;
INSERT INTO archives VALUES("$archives{$a}{'id'}", "$a", "$archives{$a}{'long_name'}",
"$archives{$a}{'nasa'}", "$archives{$a}{'url'}", "0")
EOF

                $number_of_records = $dbh->do($sql);
        }

}

###############################################################################
# build_metadata_table
###############################################################################

sub build_metadata_table {
        my ($archive) = @_;
        my $root = "$metadata_root$archive$metadata_tail";
        my ($date,%dc,$nodes,$f,@files,$tag,%ids,$id,$datestamp);
```

```
         $sql = "SELECT oaiID, dateStamp FROM metadata WHERE \
                archiveID=$archives{$archive}{'id'} ";
         $sth = $dbh->prepare($sql);
         $sth->execute;
        while (($id,$datestamp) = $sth->fetchrow_array) {
                $ids{$id} = $datestamp;
        }
         $sth->finish;

        opendir (D,"$root") || die "$!: can not open $root";
        @files = grep(!/^\./,readdir(D));
        closedir(D);

        foreach $f (@files) {

                $date = &file_date("$root/$f");

                # if the file exists, add only if the file is newer than
                # the version in the database
                if ($ids{$f}) {
                        next unless ($ids{$f} lt $date);
                        # if the file is newer than the database,
                        # delete the database copy before re-inserting
                         $dbh->do("delete from metadata where oaiID = \"$f\"");
                }

                print "adding $f\n";

                $document = $parser->parsefile ("$root/$f");
                die "no $document!!! $!:" unless ($document);

                foreach $tag (@dc_tags) {
                        $nodes = $document->getElementsByTagName($tag);
                        $n = $nodes->getLength;
                        $dc{$tag} = $nodes->toString;
                        $dc{$tag} =~ s/<$tag>//g;
                        $dc{$tag} =~ s#</$tag>$##g;
                        $dc{$tag} =~ s#</$tag>#; #g;
                        #print  $tag ." ". $dc{$tag} . "\n";
#print "string = $dc{$tag}\n";
                        $dc{$tag} = &remove_entities("$dc{$tag}");
#print "string2 = $dc{$tag}\n";
                }

$sql = <<EOF;
INSERT IGNORE INTO metadata VALUES("$f", "$date", "$archives{$archive}{'id'}",
"$dc{'title'}",
"$dc{'creator'}", "$dc{'subject'}", "$dc{'description'}", "$dc{'publisher'}",
"$dc{'contributor'}", "$dc{'date'}", "$dc{'type'}", "$dc{'format'}",
"$dc{'identifier'}", "$dc{'source'}", "$dc{'language'}", "$dc{'relation'}",
"$dc{'coverage'}", "$dc{'rights'}")
EOF

                $number_of_records = $dbh->do($sql);
                #print "sql = $sql\n";
                #$dbh->do($sql) || die "$! -- ack";
                undef %dc;
        }

}

##############################################################################
# file_date
##############################################################################

sub file_date {
        my ($file) = @_;
        my ($seconds,$month,$day,$year);

        $seconds = (stat($file)) [9];
        $year = (localtime($seconds)) [5];
        $year += 1900;

        $month = (localtime($seconds)) [4];
        $month++;       # 0 indexed
        if ($month < 10) { $month = "0$month"; }

        $day = (localtime($seconds)) [3];
        if ($day < 10) { $day = "0$day"; }

        return ("$year-$month-$day");
```

```
}

##############################################################################
# remove_entities
##############################################################################

sub remove_entities {
        my($string) = @_;

        %entities = (
                "amp"   =>      "&",
                "gt"    =>      ">",
                "lt"    =>      "<",
                "apos"  =>      "\'",
                "quot"  =>      "\'"
        );

        foreach $e (keys(%entities)) {
                $string =~ s/\&$e;/$entities{$e}/g;
        }

        return($string);

}

::::::::::::::
bin/today.pl
::::::::::::::
#!/usr/bin/perl

($sec, $min, $hour, $mday, $mon, $year, $wday, $yday, $isdst) = localtime;

$year += 1900;

$mon++;         # 0 indexed
if ($mon < 10) { $mon = "0$mon"; }
if ($mday < 10) { $mday = "0$mday"; }

print "$year-$mon-$mday\n";
::::::::::::::
bin/usage.pl
::::::::::::::
#!/usr/bin/perl

$usage_log_root = "/usr/local/web/htdocs/ntrs/ntrs/_log.pkg/usage/days";
$separator = ":::";
$nslookup = "/usr/sbin/nslookup";
$grep = "/bin/grep";

while (<>) {

        next unless (/&redirect/);
        $line = $_;
        chomp($line);

        ($host, $id, $date) = &split_log_entry($line);

        if ($hosts{"$host"}{"$date"}) {
                $hosts{"$host"}{"$date"} .= "$separator$id";
        } else {
                $hosts{"$host"}{"$date"} = "$id";
        }

        if ($ids{"$id"}{"$date"}) {
                $ids{"$id"}{"$date"} .= ":$host";
        } else {
                $ids{"$id"}{"$date"} = "$host";
        }

}

&write_days;

exit;

##############################################################################

sub split_log_entry {
        my($line) = @_;
        my($addr, $time, $url);
        my($host, $date, $id);
        my($year, $month, $day);

        %months = (     "Jan" => "01",
                        "Feb" => "02",
                        "Mar" => "03",
                        "Apr" => "04",
                        "May" => "05",
                        "Jun" => "06",
                        "Jul" => "07",
                        "Aug" => "08",
                        "Sep" => "09",
                        "Oct" => "10",
                        "Nov" => "11",
                        "Dec" => "12"
                );

        ($addr, undef, undef, $time, undef, undef, $url) = split (/ /,$line);

        # convert [28/Apr/2003:16:05:41] to 2003-04-28

        $time =~ s/\[//;
        $time =~ s/:.*//;
        ($day,$month,$year) = split('/',$time);
        $month = $months{"$month"};
        $day = "$year-$month-$day";

        # get the oaiID from the URL
        $id = $url;
        $id =~ s/^.*oaiID=//g;
        $id =~ s/ HTTP.*$//g;

        return ("$addr", "$id", "$day");
}

sub write_days {
        my ($ids);
        my (@ids);

        foreach $h (keys(%hosts)) {
                %temp = %{$hosts{$h}};
                $ip_name = `$nslookup $h | $grep Name`;
                if ($ip_name) {
                        chomp($ip_name);
                        $ip_name =~ s/Name:\s*//;
                } else {
                        # use the ip addr if the ip name isn't there
                        $ip_name = $h;
                }

                foreach $d (keys(%temp)) {
                        @ids = split(/$separator/,$temp{$d});

                        open(F,">>$usage_log_root/$d");
                        foreach $i (@ids) {
                                print "$ip_name $d $i \n";
                                print F "$ip_name $d $i \n";
                        }
                        close(F);
                }
        }

}
```

Appendix 6: Unedited Daily Progress Reports

The following status reports are included for completeness. They have not been edited for clarity, and do not represent the totality of work that was done and on what days it was done. However, they may provide insight to future NTRS administrators as to the progression of design ideas and obstacles that were addressed.

```
::::::::::::::
2003-02-25
::::::::::::::
- worked on various optimizations on the "populate-ntrs.pl" command.
  one big change was a re-structuring of the hashes to be of the form:

$archives{"naca"}{"id"} = "1";
$archives{"naca"}{"long_name"} = "National Advisory Committee for Aeronautics (NACA)";
$archives{"naca"}{"url"} = "http://naca.larc.nasa.gov/";

$archives{"ltrs"}{"id"} = "2";
$archives{"ltrs"}{"long_name"} = "NASA Langley Research Center";
$archives{"ltrs"}{"url"} = "http://techreports.larc.nasa.gov/ltrs/";

etc.

- worked on incorporating the ECD (OSTI) collection into NTRS.

- moved mpegs from ils.unc.edu to belvedere
::::::::::::::
2003-03-06
::::::::::::::
- began harvesting biomedcentral.com

- cosmetic changes to the "about" method

- added CASI ip addrs to the access control list for ntrs.nasa.gov

- fixed populate.pl (from tuesday's work).  populate.pl takes the metadata
harvested from OAI repositories and puts them in the MySQL database of
ntrs.nasa.gov

- finished moving JoAnne's mpgs to DMSS (moved from ils.unc.edu on tuesday)

- fixed cross-site scripting vulnerability in ntrs.nasa.gov
        ref: CERT Advisory CA-2000-02
        http://www.cert.org/tech_tips/malicious_code_mitigation.html/

- changed the ecd.osti.gov OAI repository to include the values that go
into dc:relation to also go into dc:type.  this is done for ease of
NTRS indexing.  began re-harvesting ecd.osti.gov

- answered questions for Juliet Pao's disaster recovery inquiry
::::::::::::::
2003-03-11
::::::::::::::
- noticed weirdness with mysql...  the database was up, but was hanging on
  queries.  I restarted the database (previous restart = 12/10/02), but it
  was not straightforward.  I downloaded mysql 4.0.11, but eventually
  restarted 4.0.5.  I'm tempted to wait until 4.1 to switch.

- deleted the OSTI Energy Citation Database records and re-added them.
  Earlier, I had modified the OSTI ECD dataprovider to provider fuller
  DC:type information, and I wanted to re-index with the new DC records.

- the biomedcentral harvest from last week had crashed.  I've resumed
  harvesting from them.  I had to modify TestHarvest.pm to not print the
  "rights" container on harvested metadata.  The DC:rights and OAI
  "about container" rights element were confusing the XML parser.

- met with George Roncaglia, Juliet Pao & JoAnne Rocker for a strategy
  session.
::::::::::::::
2003-03-13
::::::::::::::
- small mods to build-table.pl (replace "prepare"s with "do"s)
- looked for areas to optimize in index.cgi, search.pl

- sent email to oai-implementers about decommissioning the 1.1 oai interfaces

- worked with Robert Schmunk (NASA GISS) on getting their OAI interface up

- worked on the ordering method to make it context sensitive
  (nasa vs. non-nasa; printing the title, authors, etc. so users can
  have a nice page to print out, bookmark, etc.)

::::::::::::::
2003-03-17
::::::::::::::
- attended CASI security telecon

- attended ODU/NASA strategy meeting for DL futures
::::::::::::::
2003-03-27
::::::::::::::

- was re-badged at LaRC; attended saftey videos, and related logistical
tasks

- met with members of the LaRC Technical Library and advised them
with respect to Dublin Core creation, a new search engine for LTRS,
Open Archives Initiative, and other digital library strategies

- made cosmetic improvements in the NTRS "about" screen
::::::::::::::
2003-04-01
::::::::::::::
- presented (w/ JoAnne) to the STI Program Office a presentation about
NTRS, buckets and recommendation services.
       http://www.cs.odu.edu/~mln/pubs/stipo-2003-04-01.ppt

- met with George and JoAnne to discuss digital library strategies

- minor maintenance work on NTRS
::::::::::::::
2003-04-02
::::::::::::::
- joined a telecon with representatives from LaRC and JPL concerning
what JPL thought was an attack on their server from
techreports.larc.nasa.gov.  it turned out that it was a false positive
in JPL's network logs and there was no such attack.

::::::::::::::
2003-04-03
::::::::::::::
- wrote the maintainer of http://genesis2.jpl.nasa.gov/ -- I'll add that
archive next week after he completes some administrative tasks

- code to compute the total number of seconds for start-to-finish
generation of the display is now included when "&debug=on" is set
in the URL (debug mode also generates a lot of other information)

- significant speed improvements!!!

       minor attribution:
       - hardwire the preferences in &internal_read_state
       - don't call &fix_htaccess
       -  comment out unneeded subroutines
               shorten_bibfile
               append_bibfile
               lock
               unlock
               name_collision
       - reduce pagination increment from 40 to 25.
       - don't read RFC 1807 metadata if method eq search

       noticable attribution:
       - avoid initial double searching with FOUND_ROWS()
         (new in MySQL 4.0; previously the total hit size
          was computed using a slower method)

       dominate attribution:
       - turned on the query cache; currently set at 100MB.

initial queries are faster because of the FOUND_ROWS speedup, and
repeated queries are instaneous because of the query cache.
```

```
we should look at increasing the cache size when the extra memory
arrives for belvedere.larc.nasa.gov

::::::::::::::
2003-04-08
::::::::::::::
- added 128.158.* (all MSFC addrs) to the access control list

- harvested genesis2.jpl.nasa.gov

- switched NTRS's harvesting of LTRS & NACA from OAI-PMH 1.1 to OAI-PMH 2.0

- rebuilt the index to reflect the new LTRS & NACA records

- the remaining 1.1 archives are:

Aeronautical Research Council (UK)
NASA Johnson Space Center
NASA Marshall Space Flight Center
RIACS (NASA Ames Research Center)

I'm unsure of the migration plans for these archives.  They can stay at 1.1
and not really cause a problem for us.

- wrote to Russell Hope <rhope@jpl.nasa.gov> and
barbara.amargo <barbara.amargo@jpl.nasa.gov> about OAI migration for JPLTRS.

- while trying to improve the situation, I accidently blew away the tables
in the database.  the rebuild has started.  :-(

- removed the LTRS 1.1 interface.  could not remove the NACA 1.1 interface;
I've emailed Glenn Kimbrough about why I no longer have sudo/runas on
techreports.
::::::::::::::
2003-04-10
::::::::::::::
- added GENESIS (JPL) to NTRS.  they're running eprints2:
        http://genesis2.jpl.nasa.gov/

- worked on the "ordering" method
        - added the oaiID as a primary key to the metadata table
        (trick:  type "text" can't be a key, since it has no length.
         do this instead: "primary key (oaiID(255))"

        the result is a much faster lookup when you click on
        "No Digital Version Available - Order This Document"
        and get the order screen

        - ordering page is now context dependent:
                1. NASA documents now point to the CASI ordering page
                   (DOC ID is included if the metadata comes from CASI)
                2. non-NASA documents refer the user to the originating
                   archive
                3. no argument (from the footer) gives basic CASI
                   contact info.

- fixed in the update mode of the search function; the SQL statement there
did not have SQL_CALC_FOUND_ROWS added yet.
::::::::::::::
2003-04-15
::::::::::::::
- changes on the help page (as per JoAnne's email)

- added the NASA STI Help Desk to the footer

- added 2 JPL subnets to the _tc.pkg/index.tc

- harvested GISS; added GISS to NTRS interface

- cleaned up some of the file structure in the OAI 1.1 -> 2.0 migration

- optimized ListIdentifiers & ListRecords methods for the OAI interfaces
for LTRS, NACA & NTRS.  these are the same optimizations made for OSTI &
Open Video.

- added text in the "About" page to clarify that simple & advanced searches
give only NASA info by default (my solution to an issue raised in
JoAnne's email)

- worked on some graceful error handling if the database is down (unfinished)
```

```
::::::::::::::
2003-04-16
::::::::::::::
- had a telephone conversation w/ Don Marks re:
        - xfer of new format CASI records -> ODU/LaRC
        - xfer of NACA metadata+data from ODU/LaRC -> CASI
::::::::::::::
2003-04-17
::::::::::::::
- fixed a bug in which GISS identifiers were not showing up via a browse
(problem:  GSFC identifiers are currently surpressed, pattern matched only
on "NASA Goddard").  Robert Schmunk brought this to my attention.

- this fix also uncovered another bug:  in search.pl, we had both $longName
and $longname, which caused different behaivors between browsing &
searching.  fixed.

- modified populate-ntrs.pl to change how nasa vs. non-nasa archives were
handled internally

- have search.pl process doi's like:

<identifier>http://dx.doi.org/10.1175/1520-0450(2003)042&lt;0368:ROSAAG&gt;2.0.C
O;2</identifier>

I changed the split to work on "; " instead of just ";".  this preserves
the legitimate semi-colons in some URLs (esp. GISS DOIs).

the end solution is simple, but this took a lot of time to get to something
this simple.
::::::::::::::
2003-04-22
::::::::::::::
- fixed the OAI 2.0 interfaces for NACA & LTRS.  for some reason, @records
was not being set in oai/oai.pl.  I'm not sure why this failed -- it was
working as recently as 2003-04-15.

- re-harvested LTRS into NTRS.

- fixed casi2dc.pl and regenerated all the CASI metadata.  this is due to
the errors that JoAnne discovered about some metadata values being recycled
among records

- wrote a script to cull NACA metadata from the CASI metadata.  you can't
rely on dates, since some of the RMs were cleared for public release in the
1960s and 1970s.  but since we've separated NACA & NASA, we should have those
records pulled out.

- added to the "about" page a list of NTRS repositories from the previous
version that are not yet integrated with the OAI NTRS: ADS, DFRC, GRC, JPL

- edited the stopword list for MySQL 4.0.12:  took out the words "zero", "one",
"two" .. "nine"

- installed GNU make 3.80 on belvedere (needed for compiling mysql on
belvedere; the default solaris "make" command is broken)

- installed GNU m4 (the solaris version of m4 is insufficient to compile
bison)

- installed GNU bison 1.875 (the solaris version of yacc is insufficient to
compile MySQL) (bison == yacc)

- problems compiling MySQL 4.0.12.  root of the problem:  can't find
the stdc++ libs:

http://dbforums.com/arch/114/2002/8/478613

my final hack for the day:

belvedere:/usr/lib % sudo ln -s ../local/lib/libstdc++.so.2.10.0 libstdc++.so.2.10.0

belvedere:/usr/lib % sudo ln -s ../local/lib/libstdc++.a.2.10.0 libstdc++.a.2.10.0

also of note is this article favoring gcc 2.95.2 over 2.95.3:
http://www.netsys.com/cgi-bin/display_article.cgi?1227
::::::::::::::
2003-04-24
::::::::::::::
- finally compiled & installed MySQL 4.0.12.  I used the fix from:
        http://dbforums.com/arch/114/2002/8/478613

on 2003-04-22, I almost had it, but was trying to generate "gen_lex_hash.h"
instead of the correct "lex_hash.h"

- edited myisam/ft_static.c to remove "zero", "one", "two" .. "nine"
from the stopword list

- edited the startup script to be:

/usr/local/mysql/bin/mysqld_safe --user=mysql --ft_min_word_len=1 --
query_cache_size=100000000 &

ft_min_word_len used to be "3"; now any word 1 character or longer is indexed
(as long as its not in the explicit stoplist).  this means things like "TP"
and author initials should be indexed.

- started the indexing; it should complete in ~ 24 hours

- we can successfully search for:

        - "zero", "one", "two", ... "nine"
        - TP, TM, etc.


- switched all DBI->connect calls to:

  $dbh = DBI->connect($datasource, $user, $password , {'PrintError' => 0})
               || db_error("$DBI::errstr");

this prints out a nice message to the user, and then sends email to
ntrs-admin@larc.nasa.gov to let them know the database is down.
I believe just JoAnne and I are on this list now.  JoAnne owns it.
::::::::::::::
2003-04-28
::::::::::::::
- major push to get NTRS ready for public release (~ 4pm 4/28/03)

- fixed:
        - quoted searching in simple vs. advanced
        - ARC/SSC/GSFC/KSC report URL suppression (pending CASI approval)

- content changes (from JoAnne, Lynn):
        - about page
        - help page
        - linking meatbal

- scripts to JoAnne to turn on/off old & new versions of NTRS

- elapsed time of searches is now always computed; its hidden in an html
comment if debug is not set.

::::::::::::::
2003-05-06
::::::::::::::
- fixed the techreports.larc.nasa.gov/cgi-bin/NTRS script to redirect to
ntrs.nasa.gov if there are no search words, and to perform the search
if there are.  this is b/c Dryden needs to use our search capability.
the NTRS script no longer has the "Return to NTRS" line -- NACA & DFRC
will have to use the browser navigation.

- fixed naca.larc.nasa.gov to post to techreports.larc.nasa.gov/cgi-bin/NTRS
this is the same situation that DFRC had.  this is a stop-gap measure
until the new NACA server is ready.

- fixed the AND/OR'ing of fields in the advanced search interface.

- fixed the advanced search interface so that not specifying search terms
will bring back 0 records instead of bringing back all records.  bringing back
all records is expensive, and is rarely what is intended.

- moved the timing of the search function from index.cgi to search.pl
(reason: initially, we timed everything so index.cgi was the right place.
then we timed only searches, so moving it out was the right thing to do).

- added LABEL statements to the NTRS forms for 508 compliance
        - feedback page
        - help page
        - updates page
        - simple search
        - advanced search
        - browse


:::::::::::::
2003-05-07
:::::::::::::
- worked on the casi2dc.pl marc -> DC program
        - the price codes are now stored in the dc:relation field
          (probably not the right place for them, but...)
        - cosmetic improvements on how the author (dc:creator) fields
          are handled
:::::::::::::
2003-05-08
:::::::::::::
- changed the default ordering of search results from "dateStamp" to "date".
this impacts the simple search function (adv search always explicitly set
the orderby variable to "dateStamp").

- fixed the LR/LI bug pointed out by Robert Schmunk:

     if ($current_record > 1)
     { print $result; }

should be:

     if ($current_record > 0)
     { print $result; }

fixed in LTRS, NACA, NTRS and Open Video

- xfer'd new casi2dc.pl program to belvedere; regenerating metadata with
the price code in the relation field.

- changed ordering.pl to display the relation field as the "price code" if the
archive is CASI.  the price code is linked to
http://www.sti.nasa.gov/price1.pdf

- sent email to Laura re: a new banner image that can do the right thing (tm)
in a width=100% table.

- as per JoAnne's email of today, casi2dc.pl now surpresses microfiche price
codes.

- responded to Russell Hope's (JPL) email re: migrating or not migrating to
OAI-PMH for JPLTRS.


:::::::::::::
2003-05-13
:::::::::::::
- rewrote oai/oai.pl to optimize OAI-PMH harvesting.  tested some; still
needs more tuning & testing.

- updated the footer for the NACA files that are generated by naca-bib.

- the metadata indexing did not finish over the w/e because of errors in:

 oai:casi.ntrs.nasa.gov:19800007770

bad xml causes the indexer to hang.  this record had trash in the elation
field.  I've manually edited and resumed indexing.

looking in the file userserv.nasaunun1a.OUT, there is trash in the
price code field:

        <D037b,D037c,D037f(001)>|<en|

a good field should look like:

        <D037b,D037c,D037f(001)>CASI|A04|Hardcopy

I've added this field to casi2dc.pl:

        next if ($line =~ /|<en|$/);

in case there are other cases or I have to regenerate.

- fixed an error that the datestamp(s) from the weekly updates would not
be propagated past the initial page to subsequent pages.

- added SP-291 to NTRS as per JoAnne's request
```

```
:::::::::::::
2003-05-15
:::::::::::::
- interesting note:  in the about method, the "last update" section really
only counts the last new record added.  this is b/c the date is from the stat
of:

$file = "/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/$archive/metadata";

and the directory "metadata" doesn't get updated when a file is modified;
only when a file is added does it get updated.

- I created a bunch of symbolic links in:

/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest

so all of the archive name -> directory names would match.  this allows
us to map from:

oai:NASAMSFCEPRINTS:10

to:

/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/mtrs

via the symbolic link at:

/usr/local/web/htdocs/ntrs/ODL-Harvest/Harvest/NASAMSFCEPRINTS

- at the current rate of OAI harvesting delay, it would take ~ 2 days
to completely harvest NTRS:

        (550,000 records / 300 records a response) * 1.5 minutes per response
                = 2750 minutes = 45.83 hours

- using the DBM cache, we cut the time per response to ~ 16 seconds
        which is ~ 7.63 hours

ah...  but the DBM cache had many problems; including:

        - the default DBM does not support mult-dimensional hash
          (I could have installed MLDBM, but that seemed far off the critical
          path)

        - I "flattened" the multi-dimensional hash into multiple hashes,
          but hashes don't have any order, and it would be nice to have
          the ids sorted by datestamp (to enforce the weak idempotency as
          described in 3.5.1 of the OAI-PMH protocol doc).  I could sort
          the id hash by its values (which are datestamps), but that took
          a really *long* time -- longer than hitting mysql.

        - so...  I've written my own cache format of a simple text file:

                $id:::$dateStamp:::$shortName:::$subject

        - this take about 30 seconds for a cache hit, so this means a
          complete harvest in ~ 15.3 hours

        - it is the cron job's responsibility to remove the cache *after* a
          harvest.  if a harvesting session spans 2 (or more) versions of
          the cache, I believe the requirements of 3.5.1 will be in ok.

        - exception:  the cache is bypassed if we have a GetRecord request.
          its cheaper to hit the database for a single record, rather than
          slurp in the entire cache.

- although not required, we really should populate the provenance containers
:::::::::::::
2003-05-19
:::::::::::::
- cleansed some of the manual metadata (ARC, SSC) for good re-export

- arranged some of the directory structure for better re-export

- added provenance record support.  although not strictly required, it is
the right thing to do when you are aggregating.  see:

        http://www.openarchives.org/OAI/2.0/guidelines-provenance.htm

- had some problems with ListIdentifiers
```

```
::::::::::::::
2003-05-20
::::::::::::::
- reworked and optimized the code in ListRecords, ListIdentifiers, and oai.pl
everthing works a lot faster now.

- investigating using awstats:

http://awstats.sourceforge.net/
::::::::::::::
2003-06-03
::::::::::::::
- deleted some GISS ids as per Robert's email.  GISS will no longer be
exporting to NTRS pubs that are in the pre-print stage (he uses diff ids
to indicate pub status, and NTRS currently doesn't support deleted records)

- argh!!!! I blew away the database; having to reindex.

- removed a few instances of "survey" in various places

- long discussion w/ JoAnne about RDBMSs vs. IR.  mentioned that we would
eventually investigate Lucene  http://jakarta.apache.org/lucene/docs/

- added TC to the get_log method (it is now protected)

- worked with awstats ...  I'm not sure the query portion (critical for NTRS)
can be easily processed.


::::::::::::::
2003-06-05
::::::::::::::
- long discussion w/ JoAnne & George:  staff, dois, databases

- cleaning up metadata records based on Kat's input...  some are ATRS & GTRS
scraping issues, but one was bad in the MAGiC metadata, and others are
errors in the casi conversion:

        - oai:casi.ntrs.nasa.gov:20030007776 had a "&" in the type field

- in processing Kat's input, I noticed that the cache was corrupted...
could this be a race condition?  2 processes trying to write the cache at
the same time?  I don't think so, but I'm not sure.

I've made the cache writing section of &get_ids write to a temp file and
then move the temp file to the real file just in case.

- also noticed -- no empty resumptionToken on the last harvest with sets?

argh -- this needs to be changed:

                if ($current_record == $total_records) {
                        # done - quit and flag resumption
                        $next_resumption="makeMeEmpty";
                        last;
                }

this will rarely work for sets.
::::::::::::::
2003-06-10
::::::::::::::
- cleansed some MAGiC and LTRS metadata based on Kat's feedback; sent a note
 to Paul re: his data.

- ran awstats on the belvedere log, not the bucket log.  ip resolution
not working.

::::::::::::::
2003-06-11
::::::::::::::
- continued to work with awstats (must force update with "-update")

- creating .tar files for Sridhar to replicate NTRS @ ODU
::::::::::::::
2003-06-12
::::::::::::::
- lots of log rotation and preservation

- generated:
```

```
http://ntrs.nasa.gov/restricted/?method=display&pkg_name=_log.pkg&element_name=awstats

http://ntrs.nasa.gov/restricted/?method=display&pkg_name=_log.pkg&element_name=harvests

http://ntrs.nasa.gov/restricted/?method=display&pkg_name=_log.pkg&element_name=access


- cleaned up the admin interface

::::::::::::::
2003-06-17
::::::::::::::
- implemented the "delete" function in the admin page

- gave Don his login for the admin page

- created URLs for Sridhar to use to duplicate NTRS @ seven.research.odu.edu
::::::::::::::
2003-06-19
::::::::::::::
- added to the admin interface the capability to dynamically generate the
list of repositories in radio button and check box format.  these static
html fragments are then included at method invocation time by the
advanced search and browse page.  Eventually, they will be included in the
new updates page and in the admin page itself.

the files are stored as elements in the ntrs.pkg package:

http://ntrs.nasa.gov/?method=display&pkg_name=ntrs.pkg&element_name=checkbox.html
http://ntrs.nasa.gov/?method=display&pkg_name=ntrs.pkg&element_name=radio.html
::::::::::::::
2003-06-21
::::::::::::::
- wrote casixml2dc.pl to conver the MARC xml formatted records to DC.
it also checks for the presence of PDFs in the manner described by
"Phyllis L. Benson" <pbenson@sti.nasa.gov>

::::::::::::::
2003-06-26
::::::::::::::
- met with George & JoAnne to discuss deliverables, future architectures
and personnel development

- worked on the new "update" interface
::::::::::::::
2003-07-01
::::::::::::::
- attended a meeting @ the Naval Research Laboratory; discussed TRI, OAI,
NTRS, log analysis, etc.

- worked on the monthly cron log processing scripts
::::::::::::::
2003-07-03
::::::::::::::
- worked on the automation in the admin function...  specifically, added:
        - select.html is now generated by archives-html.pl
              select.html is used by the new update function (see below)
              and the admin function

        - worked on making archives harvestable from the admin page...
          I believe things work, but should test again on tuesday

        - harvest-cron is now database driven, so it will reflect new
              archives being added.  I think it works, but I'll need
              to test it some more

- worked on the new updates function that supports both "all" and
per-archive viewing.  I showed it to JoAnne and she approved, so I put
the new version in production.


::::::::::::::
2003-07-08
::::::::::::::
- deleted the duplicates from JoAnne's email mesg of June 20, 2003

- added argument parsing for $arg and $arg2 in the admin method
(no nasty stuff should sneak through) -- also, no null ids can be sent
to delete.pl now...

- delete.pl now has the option to delete an id just from the database, or
```

```
from the database + filesystem  (the admin page has been updated to
reflect this)

- edited log-cron to process, rotate and store the http access logs.  changed
the admin page to reflect the stored http access logs

- spoke in depth with JoAnne about how to use the admin interface, and what
the detailed analysis of the hosts/documents should do.

::::::::::::::
2003-07-15
::::::::::::::
- did some editing on the URL rewriting code in the display element to
support the correct viewing of the awstats outputs.  there was just plain
error, and other mods included supporting nested elments and not re-writing
urls for images.

- also changed the display code so if an "index.html" or "index.htm" file
exists in a directory listing, the file will be displayed instead of the
directory listing.

- added Jack Howard as a principal and emailed/phonemailed him with info
about how to view the output of awstats.  (also reset George's acct)

- on the admin page, uploading newly harvested metadata is now available
on a per-archive basis.

- investigated ways to continuously update the results of a script
running to the web page.  Client-pull/server-push don't seem like
the clean way for grabbing a script's STDOUT; I'll continue to
investigate.
::::::::::::::
2003-07-17
::::::::::::::
- updated some metadata for NACA reports

- added an acct for Calvin on belvedere

- discussed with JoAnne pulling NACA metadata from Galaxie

::::::::::::::
2003-07-18
::::::::::::::
- processed the metadata that Jane pulled from Galaxie
::::::::::::::
2003-07-22
::::::::::::::
- increased the capability of the admin page...  you can now add T&C files
from the interface -- adding new users & their functions can be entirely
web driven.  fixed a bug in the method "add_tc"

- wrapped the following 2 lines with the "if" in the lint method:

        if (!$no_lwp) {
                ($content_type, $modified_time, $server) = &head($baseurl);
                print "httpd version: $server\n";
        }

"&head" was defined in LWP, but we that's not always installed, so we
should have checked the value of $no_lwp

- major work for parsing the most frequent ids, hosts.  created the
basic database today.  it parses the bucket logs, and writes flat text
files in:

        _log.pkg/usage/days/YYYY-MM-DD

in the format:

ip_addr YYYY-MM-DD oaiID

129.217.198.165 2003-06-06 oai:ltrs.larc.nasa.gov:NASA-aiaa-98-0345

reports will be generated by dynamically parsing these files.

- later that night, I wrote the first draft of the "analyze" method.
many improvements remain to be done, but it turned out pretty nice.

- some working notes:

% cat total | awk '{print $3}' | uniq -c | sort -r | less
```

```
for top documents

% wc -l total

for total downloads

% cat total | awk '{print $1}' | uniq -c | sort -r | less

for top hosts

% wc -l 2003-04-30

for downloads that day
::::::::::::::
2003-07-24
::::::::::::::
- added capability to the "analyze" method.  in particular, the summary
stats now reflect the host & id patterns that are passed in.  also,
separate limits for hosts & ids are supported.

- discussed with JoAnne metrics they need to track & report
::::::::::::::
2003-07-31
::::::::::::::
- met with JoAnne and library personnel to discuss DL migration strategies

- implemented the "recommendation" method for NTRS.  this uses a server
at cs.odu.edu to get a list of 10 recommended ids back when given a single
id.

- access is not yet public (planned for next week); currently the results
can be accessed by:

        s/method=search/method=search2/;

- had to install LWP on belvedere for the recommendation method
::::::::::::::
2003-08-01
::::::::::::::
- continued work on the recommendation method, including demos to JoAnne and
coordination with Johan

- added "unique downloads" to the analyze method
::::::::::::::
2003-08-05
::::::::::::::
- set up the database table "harvests" to accomodate the history of
the growth of repositories

- worked some NACA metadata

- based on Leo's email, discovered a bug in the recommendation method.
it does not gracefully handle scenarios in which there are no recommendations
for a document
::::::::::::::
2003-08-07
::::::::::::::
- worked on the "growth" method to display information from the "harvests"
table

- baselined the harvests table with data from 2003-08-07

- integrated populating "harvests" into populate-ntrs.pl
::::::::::::::
2003-08-08
::::::::::::::
- finished the growth method after the regular 7pm harvest
::::::::::::::
2003-08-12
::::::::::::::
- fixed the growth method so dates of the form YYYY and YYYY-MM would be
promoted to full YYYY-MM-DD form so MySQL would understand them.

- the recommendation method now gracefully handles situations when
there are no recommendations

- processing NACA reports

- replaced all instances of my name on naca.larc.nasa.gov with JoAnne's
(I *think* I got all of them ;-)
```

```
this invovled re-running "create-naca -index -redo" on all the reports
::::::::::::::
2003-08-14
::::::::::::::
- Chuck Wilson added more memory to belvedere last night, but the mysql
daemon did not restart successfully.

/etc/rc2.d/S81safemysqld

did not appear to run.  I restarted it by hand today, but I'm not sure
why it did not automatically come back up.

- worked on the bucket deployment report

::::::::::::::
2003-08-15
::::::::::::::
- worked on the bucket deployment report

::::::::::::::
2003-08-19
::::::::::::::
- arranged files on belvedere to receive NACA files from techreports

- initiated transfer of files from techreports to belvdere
::::::::::::::
2003-08-21
::::::::::::::
- finished xfer of NACA files from techreports to belvedere

- discussion w/ JoAnne
::::::::::::::
2003-09-02
::::::::::::::
- fixed the log-cron.pl script (which is run monthly to process and rotate
the various http and bucket logs)

- problems discovered with the August http log - August 2 - August 13 are
missing!  I don't think my script ate them, but there is no way to tell now.
I need to periodically verify the http log to make sure it is ok.

- also, I've modified log-cron.pl to keep a copy of the previous http log
file for 1 month.

- problem uncovered:  the log rotation in log-cron.pl screws
up the logging of httpd I had to restart the daemon with
/etc/rc3.d/S50apache.orig restart

I could try to run this restart from inside the script, but I think that will
require sudo (and thus a passwd).  I've commented out the log rotation stuff
for now.
::::::::::::::
2003-09-04
::::::::::::::
- turned on the recommendation service

- fixed some refer files on LTRS (got an email from
WWW Administrator <liinwwwa@ira.uka.de> that there were problems).

- grabbed pdfs from CASI as well, but not linked yet -- holding on approval
from JoAnne.

- regenerated the data lost from the http log file from the bucket access
log file.  I focused just on days 2003-08-02 through 2003-08-13.  Days 01 and
14 surely lost data too, but figuring out where/what would be a pain,
and I'd rather be conservative than double count.

- generated a list of CASI ids for JoAnne
::::::::::::::
2003-09-11
::::::::::::::
- called Russ Woodson (JPL), emailed him and Russ Hope (also JPL)
re: OAI.
::::::::::::::
2003-09-26
::::::::::::::
- s/ntrs-admin/ntrs+admin/g

- fixed naca.larc.nasa.gov to point to NTRS (this included a hack in the
"advanced mode" of the search method to allow the "keywords" argument
```

```
to spill into title, creator, description, date, type

- added the downtime message for the upcoming system upgrade
::::::::::::::
2003-09-29
::::::::::::::
- built the bulk of the new NACA Technical Report Server; only cosmetic
changes should remain

- tested the new (old) OAI-PMH baseURL:

        http://naca-test.larc.nasa.gov/oai2.0/

works fine -- no change from the version running on techreports.

- fixed a cosmetic error in NTRS's feedback method

- fixed a mislabeled NACA report as emailed by JoAnne.
::::::::::::::
2003-09-30
::::::::::::::
- spoke with Russell Hope on the phone today re: JPL OAI

- worked on the "about" & "help" methods -- took out all NTRS references
and replace with NACA specific examples & info
```

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE *(DD-MM-YYYY)* 01- 09 - 2005	**2. REPORT TYPE** Contractor Report	**3. DATES COVERED** *(From - To)*

4. TITLE AND SUBTITLE Final Report for the Development of the NASA Technical Report Server (NTRS)	**5a. CONTRACT NUMBER** NAS1-01052
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Nelson, Michael L.	**5d. PROJECT NUMBER**
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER 23-104-05-20-30
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) NASA Langley Research Center Hampton, VA 23681-2199	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) National Aeronautics and Space Administration Washington, DC 20546-0001	**10. SPONSOR/MONITOR'S ACRONYM(S)** NASA
	11. SPONSOR/MONITOR'S REPORT NUMBER(S) NASA/CR-2005-213515

12. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Category 82
Availability: NASA CASI (301) 621-0390

13. SUPPLEMENTARY NOTES
Langley Technical Monitor: Calvin Mackey.
An electronic version can be found at http://ntrs.nasa.gov

14. ABSTRACT

The author performed a variety of research, development and consulting tasks for NASA Langley Research Center in the area of digital libraries (DLs) and supporting technologies, such as the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). In particular, the development focused on the NASA Technical Report Server (NTRS) and its transition from a distributed searching model to one that uses the OAI-PMH. The Open Archives Initiative (OAI) is an international consortium focused on furthering the interoperability of DLs through the use of "metadata harvesting". The OAI-PMH version of NTRS went into public production on April 28, 2003. Since that time, it has been extremely well received. In addition to providing the NTRS user community with a higher level of service than the previous, distributed searching version of NTRS, it has provided more insight into how the user community uses NTRS in a variety of deployment scenarios. This report details the design, implementation and maintenance of the NTRS. Source code is included in the appendices.

15. SUBJECT TERMS
Digital Library; NTRS; OAI; OAI-PMH; Federating; Harvesting; Metadata

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON**
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			STI Help Desk (email: help@sti.nasa.gov)
					19b. TELEPHONE NUMBER ***(Include area code)***
U	U	U	UU	146	(301) 621-0390